Chapter 2

Games as Experiments (1912–1977)

Electronic Computers and Games

Engineers of the 1930s and 1940s from Germany, England, and the United States independently developed electromechanical and electronic computing devices capable of performing a variety of mathematical calculations and problem-solving functions.* Early machines such as the American-built Electronic Numerical Integrator and Computer (ENIAC) and English Colossus computers calculated firing tables for artillery and deciphered encrypted messages, operations driven by the military needs of World War II. These mid-century behemoths weighed several tons, occupied entire rooms, and utilized physical switches or vacuum tubes to denote the binary computer logic states of "on" and "off." Computers such as these were almost exclusively found in specialized research labs housed in universities or government facilities. Access to them was highly restricted. In the decades after World War II, technologies became faster, smaller, and, crucially, more affordable. Small transistors replaced large vacuum tubes, creating a new class of refrigerator-sized "minicomputers." In addition to the differences in size and cost, access to minicomputers was considerably less restrictive allowing for the formation and proliferation of the computer hacker culture in universities in the 1960s and 1970s.

* Unrealized projects such as Babbage's mid-nineteenth century Analytical Engine, however, showed that the concept of general purpose computing had been in existence for at least 100 years prior.

Computer games were important within this context as they illustrated theories of artificial intelligence and provided a focus for radical innovation that was applied to other areas. Although games were particularly associated with microcomputers, few people outside of research and educational contexts had familiarity with them. The games that developed within this noncommercialized context did, however, help create a knowledge base from which many video arcade and home computer game designers drew.

Early Games in Research and Scientific Demonstration

Chess and Artificial Intelligence

One of the first, and most common, appearances of computer games occurred in research related to artificial intelligence. Chess programs were a favorite of early computer scientists and have a history with machines that stretches to at least the eighteenth century with Wolfgang von Kempelen's chess playing, "Turk" automaton hoax. In order for a computer to win against a human player, the computer needed to strategize by planning several moves in advance, constantly reevaluate its position on a changing board, and dynamically adjust to the moves of its opponent; criteria that were deemed sufficient for the demonstration of "intelligence." Chess, however, may not have been the most ideal game to test artificial intelligence. Its rules were difficult for early computers to model, and required several concessions like simpler games with limited moves and fewer choices, variants that seemed to diminish the "intelligent" behavior desired.

The use of chess as a test for artificial intelligence, however, revealed an early uneasiness on the part of computer scientists to use the most technologically sophisticated equipment for game play. Many vigorously declared that game-based computer applications were strictly for research and demonstration purposes only. Serious work demanded a serious game. Thus, they felt chess was ideal as it was associated with sophistication and intelligence rather than entertainment and fun, a cultural perception that helped legitimize the work.

One of the first truly autonomous chess-playing machines was built by Leonardo Torres y Quevedo (1852–1936), an early twentieth century Spanish inventor who specialized in the creation of intelligent electromechanical devices. Torres built "el Ajedrecista" (the chess player) in 1912 (Figure 2.1) based on his experiments of the 1890s. The device could not play a full game of chess, but it was capable of a more manageable "end game" variation that used a king and a rook controlled by the computer against a single king controlled by a human player. Each piece was plugged into a small chessboard with a metal peg, which sent a signal to the machine. Since checkmate for

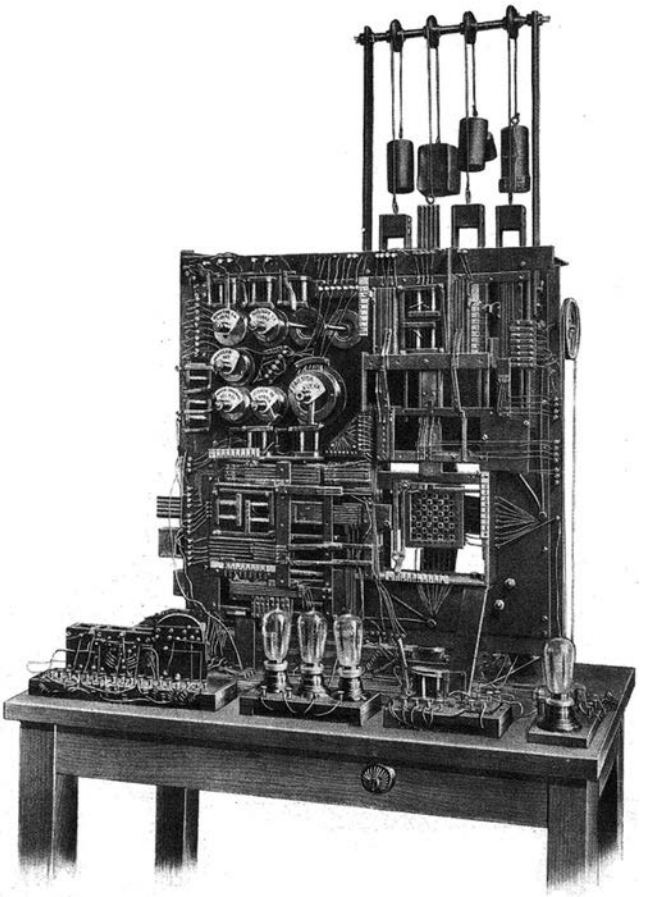

FIGURE 2.1 Leonardo Torres y Quevedo's 1912 "el Ajedrecista." (Scientific American. (1915, November 6). Torres and his remarkable automatic devices. *Scientific American*. Supplement 80. No. 2079, 296–298.)

the single human-controlled king was inevitable in this particular setup, the machine was always able to eventually beat its opponent. Torres' design, however, was still sophisticated as it signaled game states of "check" and "checkmate" to the player via a set of individual light bulbs. It could also detect and signal an illegal move. Gonzalo Torres, son of Leonardo Torres y Quevedo, created in 1920, the more refined "el Ajedrecista II" that used a phonograph to "speak" the words "check" and "checkmate" to the player. Instead of pieces that plugged into a board, it used magnets to move its pieces, as if miraculously directed by an invisible hand.

Chess continued to be used for testing the theoretical and actual capabilities of computers after World War II. In the late 1940s, Alan Turing in England and Claude Shannon in the United States designed theoretical computer programs centered on decision-making and strategy implementation, illustrated by chess. The early to mid-1950s saw limited chess programs played on early commercialized computers like the Ferranti Mark

I in England and purpose-built research computers like the Los Alamos-based Mathematical and Numerical Integrator and Computer (MANIAC). The chess program for MANIAC, for example, was only capable of playing an "anti-clerical" game of chess on a 6×6 board that excluded the diagonally moving bishops.

Computers were able to play a full game of chess without modification by the late 1950s. Although still playing at a simple skill level, the programs became more efficient and began utilizing computation-saving strategies that eliminated certain decisions with unfavorable outcomes, allowing the computer to concentrate on more productive moves. From this point on, the skill level of chess-playing programs grew greater as the number of possible decisions a computer could evaluate per second grew to the hundreds and thousands, outpacing human capabilities. Programmers looking to test the capabilities of their programs began competing against one another through computer-only chess tournaments that began at the 1970 Association for Computing Machinery (ACM) annual meeting. The culmination of these experiments was a famous 1997 chess match in which the IBM supercomputer "Deep Blue" defeated reigning chess champion Garry Kasparov.

Beyond Chess

Chess was not the only game that computer pioneers used to test theories of artificial intelligence. The game of Nim involved players taking turns at selecting a certain number of small objects from one of three heaps. By selecting a different number of objects each turn, players forced their opponent to take the last remaining object from the last pile. The game's inherent design, like chess, required planning and the ability to react to an unknown variable—the opponent's choice. The British computer company, Ferranti, used the game of Nim to demonstrate the capabilities of its "Nimrod" digital computer at the 1951 Festival of Britain's Exhibition of Science. The gigantic vacuum tube-run Nimrod, like many computers of the time, measured 12 feet wide, 5 feet tall, and 9 feet deep. Although the computer required oversight by an operator, Nimrod played Nim using a simple lighted board of circles to display the number of virtual objects left in each pile. The demonstration of this "electric brain" captivated the public. After the fair's conclusion, the Nim-playing computer was put on tour before being disassembled. Ferranti continued to manufacture computers for scientific purposes but never returned to making games.

Also in England, at the University of Cambridge, far from public display, was the Electronic Delay Storage Automatic Calculator (EDSAC). The gigantic EDSAC, built in 1947, carried out typical computer functions of the time such as calculating tables and discovering prime numbers. Its bank of three, 9-inch round cathode ray tube (CRT) displays, however, provided

the means for graphic output that gave simple, but immediate, feedback. Alexander S. Douglass' "OXO" or "Noughts and Crosses" program of 1952 used the game tic-tac-toe to illustrate his PhD thesis that related to human/computer interaction. Douglas' program used a rotary phone dial where each digit represented one of the spaces on the nine square game grid. Since the game was displayed on a CRT monitor, OXO was the first digital game to use computer-generated imagery. Douglas' involvement with computers and games, like Ferranti, was limited: he did not return to games after proving his thesis and the program was essentially forgotten. The quest to demonstrate artificial intelligence was also taken up by others such as American researcher Arthur Samuel of IBM, who created a checkers game program in 1953, as well as scientists at the Atomic Energy Laboratory at Los Alamos who, in 1954, simulated the card game, blackjack.

Turing's Imitation Game and Artificial Intelligence

Demonstrations of artificial intelligence included other game-like systems. Alan Turing, in a 1950 paper, proposed one of the most famous game-based tests for indicating artificial intelligence, the Imitation Game.[*] The first version of the theoretical game involved two participants, a male and female, who sent messages to a judge whose role was to guess their sex based solely on their responses to various questions. The game was complicated in that one of the participants was to lie about their sex, while the other was instructed to help the judge. Turing's second version of the game replaced one of the two participants with a computer. Turing reasoned, if the judge misidentified the computer as a human at the same rate that the respondent's sex was misidentified in the first game, then the computer might be considered "intelligent." Although Turing's Imitation Game generated considerable debate as a valid measure of intelligence, it nonetheless served as inspiration for computer scientists and became an important concept in the field of artificial intelligence.

In the decades after Turing, researchers created computer programs that could actually provide responses to questions. Most were unsatisfactory, but Joseph Wiezenbaum's ELIZA program, created at the Massachusetts Institute of Technology (MIT) between 1964 and 1966, was one of the first programs capable of holding a general conversation. ELIZA consisted of a language analyzer and a script that the program could reference. The most famous script of ELIZA was "doctor," which mimicked a Rogerian psychotherapist as it scanned the content of typed responses and asked passive, but leading, questions based on certain key words. Any responses that

[*] Turing, A. 1950. Computing machinery and intelligence. *Mind: A Quarterly Review of Psychology and Philosophy*, 236, 433–460. doi: 10.1093/mind/LIX.236.433.

were outside of ELIZA's limited scripts returned a generic question, hoping that it would lead the conversation to subjects more easily addressed by the computer. ELIZA's design, thus, made the user responsible for driving the conversation and required less complexity in programming than other similar experiments.

The ELIZA program and others inspired by it spread to other institutions in the United States and became, as Weizenbaum described, a "national plaything."* Wiezenbaum, however, was shocked by the unexpected responses to the program: some psychiatrists believed it could automate therapy; people using ELIZA anthropomorphized it and became highly emotional with the computer despite knowing it was not human; others claimed that ELIZA was the general solution to computers understanding human natural language. Although Wizenbaum never intended for ELIZA to have any value to psychotherapy, it initiated interest in later experiments such as PERRY, a program created in 1972 by Kenneth Colby of the Stanford Psychiatry Department, which simulated a paranoid schizophrenic. Taking the notion of computer-based conversations to their logical end, Vint Cef, a pioneer in Internet technologies then at Stanford University, later arranged a "conversation" between ELIZA and PERRY held over the ARPAnet (see below) in 1973. More entertaining than anything else, the conversation revealed the relative weaknesses of each program as ELIZA repeated the same questions over and over, while PERRY, despite simulating anger at the repetition, always responded to the questions and inevitably returned to the main topic.

Tennis for Two and the Beginning of Entertainment Applications for Computer Games

The New York-based Brookhaven National Laboratory sponsored an annual visitors' day, opening its doors for the public to learn about recent advancements in science. The goal, as part of a larger Cold War-era focus, was to inspire young people to pursue careers in science and strengthen American technological dominance. William Higinbotham, a nuclear physicist who had worked on the atomic bomb during World War II, felt that past displays at the event were unengaging and decided to create something more interactive. Higinbotham's solution was an electronic game called *Tennis for Two*, which debuted at the 1958 visitors' day (Figure 2.2). The game represented a significant departure from the previous games created within a scientific context as it did not solve specific scientific problems or demonstrate

* Weizenbaum, J. 2003. Computer power and human reason: From judgment to calculation. *The New Media Reader*, N. Wardrip-Fruin and N. Montfort (eds.), pp. 368–375, MIT Press.

FIGURE 2.2 **William Higinbotham's *Tennis for Two* at the 1958 visitors' day (second machine from the left). (Courtesy of Brookhaven National Laboratory.)**

advances in artificial intelligence: instead it entertained and inspired through its unique, competitive, two-player format.

The simple, but engaging, game featured a tennis match, seen from a sideway perspective with a vertical line representing the net and a horizontal line representing the playing field. *Tennis for Two* was based on an analog computer able to simulate the trajectory of ballistic missiles and other types of objects. The setup used a 9-inch oscilloscope for a display and featured two custom-built, single button, rotary-dial controllers. Players adjusted the trajectory of a ball through the knob controller and sent it to their opponent with a press of the controller's single button. Each time the player volleyed the ball, the computer's electromechanical switches created a loud clacking noise, which, although unintended, provided an element of sound feedback for the virtual actions. Players took turns batting the ball back and forth, trying to force their opponent to hit the ball at an awkward angle.

The game was inspired by and named after the sport of tennis, but it did not actually simulate a full game, much like earlier chess-playing games. In *Tennis for Two,* players would never miss if they pressed the controller's button while the ball was on their side of the court; effectively players existed everywhere simultaneously. This inherent violation of reality led to an experience that was only possible in an electronic format; a concept that was explored to a greater extent in the games created by computer hackers in the 1960s and 1970s. *Tennis for Two* reappeared the following year before the

EARLY EXPERIMENTS IN AUGMENTED AND VIRTUAL REALITY

Ivan Sutherland, in a famous 1965 essay entitled *The Ultimate Display*,* speculated on a number of display and control schemes for computers including non-vector, "filled" images made of colored areas and computers capable of being controlled by eye movement. Taking these concepts to their theoretical extreme, Sutherland described the ultimate display as a computer that could control matter in a room leading to "[a] chair displayed in such a room would be good enough to sit in. Handcuffs displayed in such a room would be confining, and a bullet displayed in such a room would be fatal."

Although the ultimate display was entirely theoretical as it literally created "virtual reality," Sutherland and others, nonetheless, pursued the simulation of spaces using the rapidly advancing computer technologies of the late 1960s and 1970s. Through a series of individual experiments and improvements, Sutherland, along with several student assistants, created the first head-mounted display (HMD) to show 3D objects in 1968. Using a wireframe vector display that drew straight lines from a series of coordinates, the HMD produced a simple 3D cube overlaid on the physical environment, which allowed users to see a computer-generated object in space. Sutherland's HMD coevolved with his theories of kinetic computer control introduced in *The Ultimate Display,* as the user adjusted the perspective of the image by turning his or her head. The HMD was attached to a mechanical arm suspended from the ceiling of the lab, which both tracked head movement and helped alleviate strain on the user's neck caused by the unit's considerable weight. Its intimidating appearance and the precarious nature of the setup earned the HMD the nickname, the "Sword of Damocles," an allusion to the ancient Roman parable involving a sword suspended by a single horsehair over a man's head. Sutherland's "Sword of Damocles" became a major source for later research into both virtual reality and augmented reality that would emerge in the consumer realm in the later 1980s and reemerge in the early 2010s (see Chapter 8).

* Sutherland, I. 1965. The ultimate display. In *Information processing 1965: Proceedings of IFIP Congress 65*, W. A. Kalenich (Ed.), Vol. 2, pp. 506–508, London: Macmillan and Co.

computer powering it, like many early game-playing computers, was disassembled for parts.

A modified version of *Tennis for Two* under the name "*Computer Tennis*," however, was created and displayed at the 1961 visitors' day.

The Hacker Ethic and Games

"Hacking" is a form of experimentation and solution-finding centered on cycles of creating, deconstructing, and tinkering that naturally lent itself to expression on programming electronic computers. Although highly technical, hacking was an art among early programmers, as computers were made to perform in new, often unexpected ways. The desire to iterate and

improve on programs led to a vibrant, collaborative hacking scene that originated in the later 1950s, after behemoth-sized electronic computers found their way into university labs and research institutes. Creating games was a logical extension of these ideas. Hackers explored the technical and artistic capacities of computers through hands-on experimentation in the ultimate "sandbox" environment. Although earlier engineers and computer programmers used games, such as chess, in the course of their research to test various theories and procedures, hacking was born of and motivated by a different mindset, one that extended into a system of values.

Steven Levy's 1984 book, *Hackers: Heroes of the Computer Revolution*, (Levy, 2010)* chronicled the contributions of individual hackers active between the 1950s through the early 1980s who helped build the knowledge base that led to the home computer revolution (see Chapter 6). Levy's understanding of the attitude of these individuals led him to formulate a set of hacker principles known as "the hacker ethic." The hacker ethic, as Levy outlines, is a set of beliefs and values governing people, their relationship to computers, and relationships to each other. Key concepts include the unrestricted sharing of information, a mistrust of authority, the imperative to demonstrate one's abilities, the prospect that computers are capable of creating art and beauty, and that computers can lead to the betterment of mankind.

Much of the hacker ethic was derived from the practices surrounding computers in the immediate post-World War II era. A person who wanted to run a program or compute a mathematical equation on a computer was unable to directly do so as early computers required specialized technicians to run them due to their complexity and relative fragility. As such access was strictly limited to a few. This hierarchical "priesthood" of technicians surrounding the operation of computers was intolerable for individuals driven to understand how computers worked and what their capabilities were. Many of the underlying attitudes of the hacker ethic also comingled with the counter-culture movement of the 1960s and 1970s in the United States. Hackers in and around the California computer scene and the developing Silicon Valley, were occasionally described as having long hair and strong antiwar views. Although not all hackers fully articulated or even subscribed to these ideas, this unspoken attitude was an underlying presence in many of the games produced in the 1960s and 1970s.

The Spread and Modification of *Spacewar!*

The MIT in Cambridge, Massachusetts became one of the earliest havens for computer hackers of the late 1950s and early 1960s. During late night hours, when computers were not reserved for "real" research applications, the unorthodox and curious computer science students of MIT engaged in their own,

* Levey, S. 2010. *Hackers: Heroes of the Computer Revolution*. Sebastopol, CA: O'Reilly Media, Inc.

more playful form of research. They built their own assembler and debugger programs, which allowed them to engage in more experimental endeavors such as programs that played music as well as various "display hacks." Display hacks took advantage of the visual output capabilities of newer computers through a CRT display. Hackers devised everything from noninteractive dynamic imagery to user-created mazes populated by computer-controlled mice. MIT student Steve "Slug" Russell, along with Martin Graetz and Wayne Witaenem, looking to create an impressive display hack, decided on a game.

Russell, Graetz, and Witaenem were members of a fictional research lab called the Hingham Institute, named after their Hingham Street apartment—a mocking reference to the grandiose air of buildings at Harvard University, MIT's Cambridge rival. The group's interest in space operas filled with battles between space ships, written by science fiction novelist, Edward Elmer "Doc" Smith, led them toward a space-based game and the formation of the tongue-in-cheek, Hingham Institute Study Group on Space Warfare. While searching for ways to implement the idea of outer space battles into a game, the group formulated the first articulated theory of video game design, the Hingham Institute Study Group on Space Warfare's Theory of Computer Toys, which read as follows:

1. It should demonstrate as many of the computer's resources as possible, and tax those resources to the limit
2. Within a consistent framework, it should be interesting, which means that every run should be different
3. It should make the viewer a participant[*]

Guided by these principles, Steve Russell began to program *Spacewar!* in late 1961. The game was written on the Programmed Data Processor-1 (PDP-1), a minicomputer with a CRT display manufactured by Digital Equipment Corporation (DEC). Russell's progress on the game was slow and prone to long periods of inactivity, as the code to manipulate the ships was difficult to create. Fellow MIT hacker Alan Kotok, through his connection to programmers at DEC, was able to acquire the code necessary to calculate the ship's motions. Significant progress was then made and the initial version of *Spacewar!* was completed in early 1962.

Spacewar! was a dueling game in which two players attempted to shoot the other's wedge-shaped or needle-like spaceship. Players could rotate each ship in a clockwise or counter-clockwise circle and initiate a thrust of the engines using toggle switches on the control panel of the PDP-1. The first version of *Spacewar!* took place in outer space with a few random dots representing stars and featured inertia: ships continued to glide through the screen space even when the engines were not firing (Figure 2.3).

[*] Graetz, J. M. 1981. The origin of spacewar. *Creative Computing*, 5, 61.

FIGURE 2.3 ***Spacewar!*** **played on a PDP-1 minicomputer. The trails of green and yellow left by the ships in the image are phosphorescent artifacts of the PDP-1's CRT scope. (Photo by Joi Ito. CC BY 2.0. Level adjustment from original.)**

The program spawned a flurry of activity among hackers at MIT, as each added to or modified both the game design and its visuals. Peter Samson replaced the random background stars with astronomically accurate star positions that slowly rotated; Dan Edwards added a large star in the center of the playfield, which exerted a constant gravitational tug on the ships and drew them toward destruction if they collided; Martin Graetz added a limited-use hyperspace feature that randomly placed the ship in a new position, one that had the possibility of landing the player in the path of the flying space torpedoes or next to the star—a mechanic accompanied by the spectacular visual effect of a "warp-induced photonic stress emission."[*] Further, the hyperspace mechanic became unstable overtime and had the potential to destroy a player's ship. Alan Kotok and Bob Saunders addressed the problem of skinned elbows developed from the excited gameplay on the machine's toggle switches by creating two controllers from requisitioned parts. The wooden, box-like controller consisted of two switches that regulated the rotation and amount of thrust of the ships, as well as a single button used to fire. The controller triggered the hyperspace jump through pushing the thrust lever forward and releasing. Later versions of the controller were "enhanced" with feedback in the form of an electric shock on player death.

[*] Graetz, J. M. 1981. The origin of spacewar. *Creative Computing*, 5, 66.

Spacewar! spread to universities, research labs, and commercial companies throughout the 1960s and early 1970s. Steve Russell himself ported *Spacewar!* from the PDP-1 to the PDP-6 in 1966 after arriving at Stanford University's newly founded Artificial Intelligence Laboratory, while other hackers created versions for other computers. Following the tenets of the hacker ethic, the ports of *Spacewar!* were further modified by later users with new mechanics that resulted in significantly different gameplay. A 1972 version of *Spacewar!* modified by Ralph Gorin and played at the Stanford Artificial Intelligence Laboratory (SAIL) featured five simultaneous players, space mines, partial damage, and two torpedo tubes. Just as at MIT, the hackers of SAIL created their own controllers. Other variants of the game included a "2 ½-D" version of *Spacewar!* in which the entire playfield was presented from a first person perspective—a setup that required the use of two CRT scopes, one for each player.

In addition to physically sharing the paper tape code for games like *Spacewar!*, DEC itself, the manufacturer of the PDP-1 computer, facilitated the distribution of games. *Spacewar!* for example, was used as a final testing tool for the PDP-1 and was left in the computer's memory during shipping. When the PDP-1 arrived at its destination, the game was initialized as a check to ensure that the delicate computer had arrived intact. In 1961, a group of computer users representing various technology companies and university research labs formed the Digital Equipment Computer Users Society (DECUS) with the goal of creating a community knowledge base for computer programming. Principally important was the program library established by the DECUS, which allowed its members free access to everything from utility programs to games. The DECUS library, through the 1970s, distributed demonstration packages that, in addition to the *ELIZA* program, included small collections of games such as chess, word games, tic-tac-toe, simulations of board games like *Monopoly* and *Battleship* and *Spacewar!*, among others.

Computer Networks and Games

The advent of minicomputers like the PDP-1 led to important developments in computer science that helped define the way computers are used and experienced today. In the early 1960s, minicomputers adopted a multiple user model of computing called *time-sharing* that replaced the *batch processing* model of the 1950s. Time-sharing systems used a single central computer that powered a number of user terminals. Programs were centrally run on the computer, while the terminals provided input as well as output via a visual display or through a printer. As opposed to the earlier batch processing, which consisted of a single user's program intermittently using computer resources, time-sharing allotted a portion of processing power to each terminal. With multiple users working simultaneously, the

time-sharing model created continuous activity on the computer and minimized any down time. The demand for computers grew throughout the 1960s and 1970s, with time-sharing systems proliferating at universities, research institutions, and government agencies.

The ARPAnet

The Advanced Research Products Agency (ARPA), created in 1958, was a special organization within the United States Department of Defense tasked with developing new science and technology. Initially focused on missile and space technologies, ARPA became increasingly involved with developing computer technology in the 1960s and helped support the building of time-sharing networks at MIT, UC Berkley, and the Systems Development Corporation. Each site was connected to ARPA via an individual terminal in order to easily share research advancements and facilitate communication. Seeing the growth of these and other small, local time-sharing networks, Bob Taylor, then head of the Information Processing Techniques Office at ARPA, began a project in 1966 to link individual networks together. The project eventually became known as the ARPAnet.

The initial phase of the ARPAnet was completed in 1969 and linked four time-sharing networks at UC Santa Barbara, UCLA, Stratford Research Institute, and the University of Utah, allowing users to access multiple computer mainframes. By 1971, the ARPAnet linked more than 20 locations, connecting the east and west coasts of the United States directly as well as through a major link at the University of Illinois at Urbana-Champaign. Two years later the ARPAnet had grown to 30 locations including international connections to University College of London and the Royal Radar Establishment in Norway. By 1975, the number of locations grew 61 and continued to grow exponentially as the ARPAnet infrastructure became increasingly known as the Internet, a network of networks connecting computers. The software-based World Wide Web, however, did not develop until the late 1980s to early 1990s, with commercial forces powering much of its development.

Programmed Logic for Automated Teaching Operations and Multiplayer Games

One of the largest time-sharing systems was Programmed Logic for Automated Teaching Operations (PLATO), centered at the University of Illinois at Urbana-Champaign. From its 1961 conception, PLATO was intended as a low-cost way to deliver automated educational content to students from kindergarten to college, using computer terminals. PLATO evolved rapidly from the late 1960s to early 1970s. Its TUTOR programming language allowed instructors to easily formulate lessons with interactive features. A new generation of terminals for the 1972 PLATO IV system

featured highly responsive, touch screen plasma panel displays capable of producing sound and high quality visuals in monochrome orange. Email, instant message, chat rooms, and a host of other virtual communication tools were developed for PLATO, creating the first true online community and a model of things to come with the World Wide Web.

The open environment associated with PLATO encouraged experimentation leading some instructors to develop early educational games. Paul Tenczar, who wrote the TUTOR programming language for PLATO, created two simple programs to teach concepts of computer programming. In them, animated figures were instructed to pick up objects on the screen. Bonnie Seiler's elementary mathematics game *How the West Was One + Three x Four* (1971) was particularly popular and resulted in continued development for other computer platforms through the 1990s. Instructors, however, were not the only content creators on the PLATO system, as students from a range of ages used the responsiveness of the system and high-resolution displays to create animations as well as their own games.

The PLATO network's inherent connectivity between users led to a number of games designed specifically for multiple players. One of the first multiplayer games on PLATO was a version of *Spacewar!* written in the late 1960s. Another was a version of chess. Although *Spacewar!* and chess were played between two players, later games in the 1970s incorporated a larger number of players as amateur programmers began to explore the system's capabilities. John Daleske's 1973 *Empire* grew from a project for an education class while he was attending Iowa State University. The game, unlike others on the PLATO system, allowed for eight players. Each represented one of the races from the television series, *Star Trek,* in a strategic game involving the management of a planet's economy and population, as well as directing space ships to conduct trade and diplomacy. Daleske, looking to add more action, redesigned *Empire* later in 1973 to exclusively focus on *Spacewar!*-like battles across a larger multiscreen game space. Players used a variety of keyboard commands, independent of each other, to adjust the ship's heading and direction of fire. Daleske continued to modify the game through 1981. He eventually combined the strategic elements of the first game with the space combat of the second, allowing a wide range of roles for up to 50 players, rivaling the complexity of later commercialized online games. Development of more complex versions of *Empire* continued. Each new version added and refined features based on Daleske's vision as well as suggestions from programmers and players, making it one of the earliest community-driven multiplayer games.

The code and concept of *Empire* spawned a number of network-based games developed by other individuals. Silas Warner was a well-respected and prolific PLATO programmer at Indiana University responsible for the creation of a number of games on PLATO. One of his early projects was the 1973 strategy game *Conquest*, developed from the original strategic version

of *Empire*. Other *Empire*-based programs included various "Trek" titled games of the 1980s that retained *Empire*'s original Star Trek alien races. Perhaps, the best known descendent of the "Trek" titled *Empire* games was *Netrek*, a game designed in the late 1980s and popular on university campuses through the early 1990s.

Adapting Dungeons & Dragons *to PLATO*

PLATO also featured a number of games inspired by the tabletop roleplaying game, *Dungeons & Dragons*. Heavily based on themes from J.R.R. Tolkien's *Lord of the Rings* book trilogy and game systems from tabletop war gaming, *Dungeons & Dragons* was created in 1974 by Gary Gygax of Wisconsin and Dave Arneson of Minnesota. Unlike other games both physical and digital, play of *Dungeons & Dragons* began with the creation of a character. Players chose from a list of classes/professions, such as fighter, wizard, or thief, and determined the character's physical and mental attributes by rolling dice (Figure 2.4). The setup of play was unique relative to other games, as a group of players worked together to complete episodic adventures that continued from session to session. Characters created by the players grew and changed overtime, becoming stronger and gaining new abilities. A crucial component of *Dungeons & Dragons* was the player who served as the Dungeon Master, a storyteller/director who provided information to the other players about the game world and its characters. Players interacted with the fantasy world in a variety of ways, many of which required rolling dice to simulate outcomes beyond player control.

The PLATO computer network was particularly well-suited for the adaption of games like *Dungeons & Dragons*. The mathematical systems of probability and random number generation were easily ported to computers while the social gameplay involving a group of players could rely on PLATO's well-developed communication capabilities and vibrant

FIGURE 2.4 Typical types of dice used to generate ability scores and simulate event outcomes in *Dungeons & Dragons*.

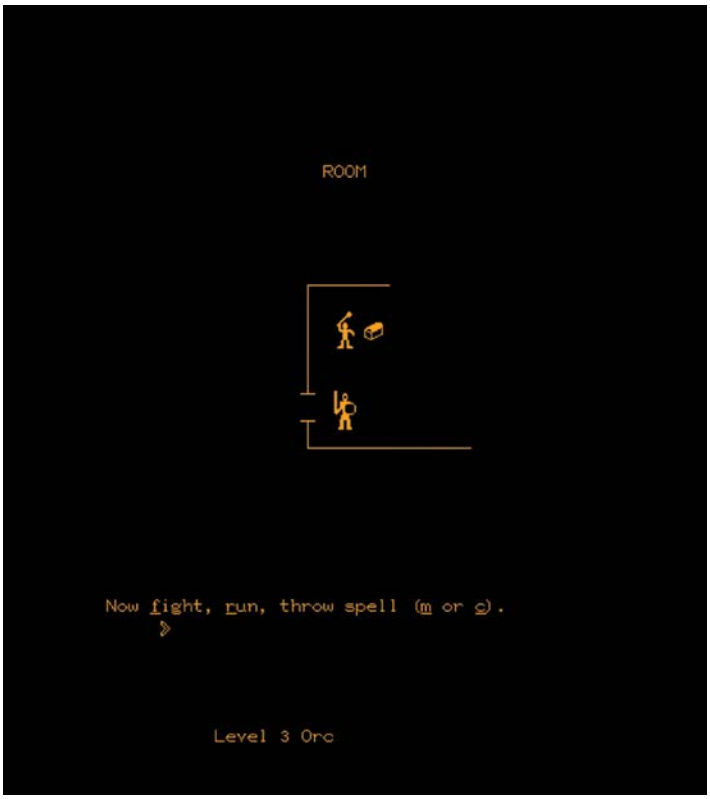

FIGURE 2.5 Encounter in *pedit5*/*The Dungeon*. (Courtesy of Paul Resch and Cyber1.org.)

community. One of the first PLATO games inspired by *Dungeons & Dragons* was a single player game called *pedit5*[*], created in 1975 by Rusty Rutherford of the University of Illinois at Urbana-Champaign (Figure 2.5). The gameplay of *pedit5* involved navigating a maze-like dungeon space of 40–50 rooms, with randomly generated locations that housed monsters and treasure. The game's ultimate goal was to accumulate 20,000 experience points, which, like *Dungeons & Dragons*, was gained through combat with monsters and collecting treasure.

Not all *Dungeons & Dragons* systems were fully replicated in *pedit5*, but it did use the game's core elements such as experience points and monster levels. Rather than distinct classes, as in *Dungeons & Dragons*, players played a character that combined the attributes of the warrior, mage, and cleric. *Pedit5* was widely played on the PLATO system, but it was unreliable because system administrators constantly deleted it to free memory resources for more "academic" applications, forcing Rutherford to rewrite the program each time.

[*] "Pedit5" was the original file name for the program and intentionally kept nondescript to prevent it from being deleted. It was also known as *The Dungeon*.

Gary Wisenhunt and Ray Wood of Southern Illinois University, tired of the erratic availability of *pedit5*, decided to write their own dungeon-based game called, *The Game of Dungeons*, better known as *dnd*.* The likelihood of *dnd* being deleted was slim, as Wisenhunt and Wood were PLATO system administrators at SIU. Relative to *pedit5*, *dnd* was more expansive. Players could venture into multiple dungeon levels, each featuring monsters of increasing difficulty. As players progressed, they ultimately encountered the "boss" character in the form of a dragon, one of the first instances of this staple element in digital games. Boss characters are particularly strong adversaries with different behaviors requiring the player to abruptly change tactics. This punctuation in gameplay was a signal in *dnd* to indicate the player was close to completing the game.

As with *Empire*, the player community was involved in the development of *dnd*. Features and improvements to the game were constantly considered and implemented. Part of the game's appeal, in addition to its quirky humor, was the inclusion of a visual maze editor, a rare feature for games of the time. It allowed both the designers and the members of the PLATO community to quickly and easily create their own dungeons. Wisenhunt and Wood eventually handed off further development of *dnd* to the brothers Dirk and Flint Pellett, who had previously created a number of add-ons and other features to the game. Another *pedit5*-inspired game was *Orthanc*, created in 1975 by Paul Resch, Larry Kemp, and Eric

DESIGNING FOR EFFICIENCY

PLATO terminals were connected through a phone line-based network capable of transmitting 1200 bits per second out and 300 bits per second in. Although PLATO was significantly faster than other network connections of the time and the terminals themselves stored text data, games needed to minimize the amount of data transmitted during gameplay to ensure a playable speed. *Pedit5* and *Oubliette*, for example, sent the player only the minimum amount of information necessary to play: i.e. the lines making the walls of the dungeon maze and the title of the space. Information about player health, player stats, or monsters was only transmitted and displayed when absolutely necessary. Thus, the screen was largely empty, but gameplay remained responsive. The designers of *Orthanc*, however, grew tired of constantly needing to request information in *pedit5* and designed the game's interface to constantly display all relevant information. To keep the gameplay fast, the entire screen was drawn once and then kept static. The game then updated individual pieces of the screen as needed, with the maze receiving the most frequent refreshes.

* Not to be confused with the very similar mainframe game, *DND* written in the BASIC programming language by Daniel Lawrence of Purdue University in the late 1970s, nor the DOS game, *DND*, developed by Bill Knight of R O Software in 1984.

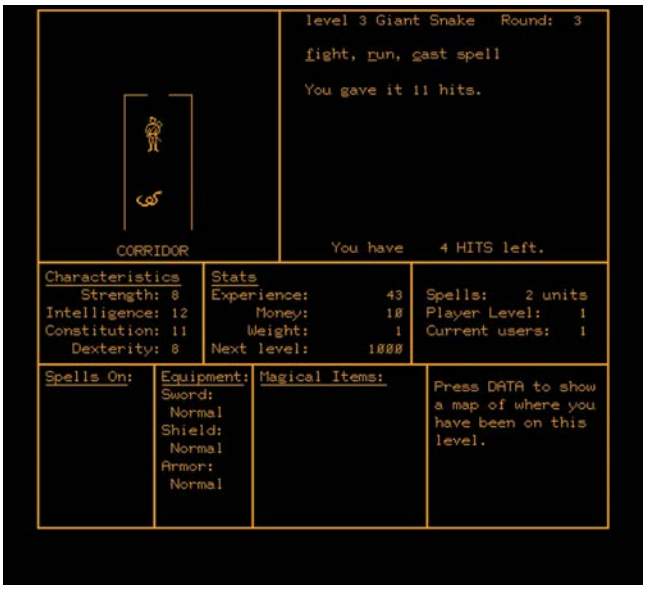

FIGURE 2.6 Traveling through the dungeon in *Orthanc*. The reference to the map in the lower right corner of the screen was added to the game much later during its continuous development. (Courtesy of Paul Resch and Cyber1.org.)

Hagstrom of the University of Illinois at Urbana-Champaign (Figure 2.6). *Orthanc* included elements intended for *pedit5* but never implemented, such as multiple level dungeons as well as certain improvements like the ability to choose a different randomized starting character if the first was undesirable.

Other dungeon exploration games on PLATO granted players the ability to form a party and play as a group. The 1975 game *Moria*,* by Kevet Duncombe and Jim Battin of the University of Illinois at Urbana-Champaign, was distinctive as one of the first multiplayer games with a *persistent game world*, where events and actions continued to take place even when the player was not playing. *Moria* also moved away from the top-down mazes of earlier dungeon-based games and replaced them with line drawings of rooms in a first person perspective. Another major distinction was the game's systems. According to *Moria*'s authors, the game was created not from *Dungeons & Dragons*, but from talking about design problems encountered in the formation of *dnd* with its creators. This allowed a greater degree of freedom in design, as the game lacked many *Dungeons & Dragons*-inspired mechanics, such as experience points and character levels to measure growth, leading to more innovative mechanics such as character progression through practicing certain skill-related actions.

* Not to be confused with the commercial release of the 1988 game *Moria*, a roguelike dungeon game (see Chapter 6).

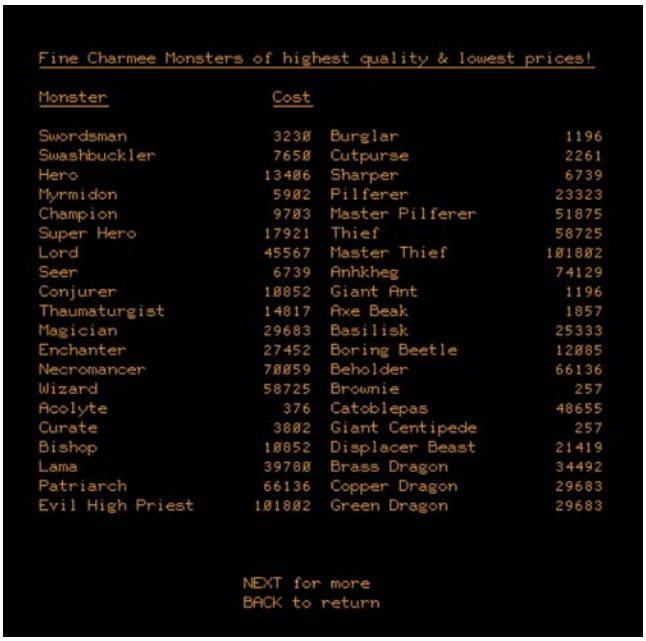

FIGURE 2.7 Selections from the Monster Store in *Oubliette*. (Courtesy of Jim Schwaiger and Cyber1.org.)

Oubliette (1977) was one of the most developed and popular roleplaying games on the PLATO system. The game was originally designed by Jim Schwaiger along with John Gaby and Bancherd DeLong out of a desire to automate the tedious dice rolling associated with *Dungeons & Dragons*. The scope of *Oubliette* was massive. Players were able to choose form 15 character races and 15 classes depending on ability scores. Prior to venturing into the game's dungeon, players roamed the streets of a large castle and shopped in multiple stores, outfitting their characters with everything from simple items to monster companions (Figure 2.7). Players could join various profession-based guilds, gamble, deposit money in banks, and socialize with other players in a chat room-like tavern. *Oubliette* was also extremely difficult for single players as the game was designed for parties of adventurers. The dungeon was full of traps and secret doors. Monsters were frequently encountered in large numbers (Figure 2.8) and death was permanent unless another player found and retrieved one's body from the dungeon.

The combination of multiplayer gameplay, robust communication capabilities, a developed game economy, character generation choices, and spell systems of *Oubliette* and the later attempt to outdo *Oubliette*, *Avatar* (1979), were not surpassed in commercialized computer games until the 1990s and 2000s when Internet infrastructure allowed visually complex massive multiplayer online roleplaying games. *Moria*, *Oubliette*, and *Avatar* established many of the design, gameplay, and keyboard command conventions seen in

FIGURE 2.8 **Encounter with seven mediums in *Oubliette*. (Courtesy of Jim Schwaiger and Cyber1.org.)**

EARLY MONETARY TRANSACTIONS

Games on PLATO occupied resources intended for educational and research purposes. This created some misgiving, since the microbiology department of the University of Illinois at Urbana-Champaign sponsored the space for *Oubliette*. *Oubliette* creator Jim Schwaiger, thus devised one of the first monetization strategies for multiplayer online games in which he "paid" the microbiology department for the game space using funds collected by the players. Players were free to play the game, but if they desired to keep their characters through the regular system purges, they would need to pay a fee of $3 per year. This unprecedented notion was met with criticism by the player base, however, those who had invested a great amount of time into their characters feared losing their progress and paid the fee. Other monetary exchanges in *Oubliette* centered on an underground economy in which players sold high-level characters and rare in-game magic items for real money in amounts in excess of $100. Although this phenomenon is typically associated with more recent massively multiplayer online role-playing games (MMORPGs), games like *Oubliette* provided one of the earliest examples of these transactions.

later commercialized computer roleplaying games. In particular, games like *Wizardry: Proving Grounds of the Mad Overlord* (Sir-tech, 1981) drew significantly from *Oubliette* and became influential in their own right by shaping the development of computer roleplaying games in the 1980s (see Chapter 6).

Early 3D and Networked Games

The early 1960s saw the first experiments that involved 3D computer graphics to illustrate scientific work. One of the first examples was a 1963 computer-generated animation by Edward Zajac of Bell Laboratories showing the movement of a rectangular satellite in space around a wireframe sphere. Each minute of animation took the computer 3–8 minutes to compute, so playback was not done on the computer but was printed to film, frame by frame. Computers were able to compute and animate wireframe images in real time and present them on CRT displays by the early 1970s.

Hackers not only jumped at the opportunity to explore the capabilities of 3D imagery through games, but also some did so in conjunction with emerging computer networks. The 1973 game, *Maze War*, or simply *Maze*, developed from a project in which Steve Colley, a high school intern at the NASA Ames Research Center, desired to create a program to utilize hidden line removal on a rotating wireframe 3D cube. Colley, assisted by fellow high school intern, Howard Palmer, elaborated on the original idea by creating a maze that was explored from a first person perspective. Players moved through the game space one square at a time, making right angle turns, looking for the exit. Another high school intern, Greg Thompson, helped add more action to the game by creating a multiplayer version that linked two Imlac PDS-1 computers, with each player represented by a large eyeball avatar. Players had the ability to shoot each other in a manner that predated the "Deathmatch" multiplayer modes of first person shooters in the 1990s (see Chapter 8).

Maze War accompanied Greg Thompson to college at MIT in 1974. There, Thompson and Dave Liebling, a member of MIT's Dynamic Modeling Group, co-created a version of the game that was playable over the ARPAnet and allowed more simultaneous players. Although the initial ARPAnet-based gameplay suffered from high latency, the problem was eventually remedied.

The PLATO system also featured games that used wireframe 3D images with multiplayer capabilities. Most important was Jim Bowery's 1974 *Spasim* (pronounced "space sim"), a game in which 32 players piloted space ships from a first person perspective and fought in 3D space. *Spasim*'s unique 3D perspective was based on an earlier program co-written by computer image pioneer, Ron Resch, that Bowery obtained while studying at the University of Iowa. Using Resch's program as a basis, Bowery crafted *Spasim* to be a 3D version of *Empire*. Players maneuvered through space using both polar and Cartesian coordinate systems, a feature that Bowery used to justify the game's presence on the network because of its educational nature. Later

in 1974, Bowery deleted the initial competition-focused version of *Spasim* and replaced it with a more cooperative game that discouraged warfare and instead, focused on the strategic management of resources in order to stabilize a planet. Like *Spacewar!* and other early computer games of the 1960s and 1970s, Bowery distributed the code for *Spasim* to other hackers who modified it to create their own games, resulting in a number of other 3D games on PLATO and beyond. Silas Warner modified *Spasim* into the 3D airplane simulator, *Airace*, which, in turn, led to the 3D combat-oriented *Airfight* of 1974 by Brand Fortner. *Spasim*'s code was also the basis for the 1975 first person tank game, *Panzer,* by John Daleske and Derek Ward.

Into the Commercial Realm

To many hackers of the 1960s and 1970s, games like *Spacewar!* were a means to an end, an interactive learning project that challenged their abilities and furthered their knowledge about computers. A number of these programmers, however, did pursue games as a profession. Silas Warner, was part of the first wave of programmers to create commercialized games for early home computers. His most noted works were those he designed and created under Muse Software: *Castle Wolfenstein* (1981) and *Beyond Castle Wolfenstein* (1983). Warner executed the programming for the Amiga version of the submarine simulation game, *Silent Service* (1986, MicroProse) as well as other contributions to console games of the early 1990s. Dave Liebling helped create the text-based interactive fiction game *Zork* while still in school, which eventually became a flagship commercial franchise of Infocom, a company he helped found (see Chapter 6). *Orthanc* programmer Paul Resch briefly worked for Atari's coin-op and home computer divisions just prior to the North American video game industry crash (see Chapter 5), while *Oubliette* creators Jim Schwaiger and John Gaby eventually entered the mobile game market (see Chapter 9). As discussed in Chapters 3 and 6, some of the earliest video arcade and home computer games of the 1970s were either inspired by or directly adapted from these experiments. Thus, the collective work of early computer hackers was crucial in establishing the commercial digital game industry.