Engineering Spore

By **David Kushner**

PHOTO-ILLUSTRATION: SEAN MCCABE; ORIGINAL PHOTO: RYAN ANSON/AFP /GETTY IMAGES

It's time to make a creature. Let's start with its body. Stretch down a pair of legs and pull out two arms so that it looks long and lean. On one end of the body, pop on intelligent eyes behind large round glasses. Add a mop of peppery hair and a prominent nose and ears. Sprinkle on scruffy semibeard growth. Call it Will Wright.

Now put the creature in its habitat: the workspace of a computer game developer in Emeryville, Calif. Spread the studios of Maxis Software, which Wright cofounded in 1987, over the floors of two nondescript office buildings. Sprinkle the interior with dirt-encrusted mountain bikes leaning against cubicle walls and overgrown, pumpkin-orange beanbag chairs. Now surround Wright with others of his kind, hunched behind desks, typing at keyboards, PC monitors glowing.

The software engineers, artists, and others who work at Maxis, now owned by video-game giant Electronic Arts, have migrated here because Wright is a legend. Over the past two decades, the 48-year-old Wright, who studied architecture and mechanical engineering at Louisiana Tech University, has utterly transformed his industry with hits like *SimCity*, in which players build virtual towns, and the best-selling computer game franchise of all time, *The Sims*, in which players create virtual people and then watch them interact. In the process, Wright has helped forge a new, more toylike frontier in computer gaming, where the main goal is not so much to score points or kill bad guys but to create cool stuff.

The game they're working on this bright February day is called *Spore*, and it's the most ridiculously ambitious simulation game yet. Sure, there've been virtual worlds like *Second Life*, which let you customize your characters. In games like *Fable* and *Black & White*, the characters even evolve in appearance and reputation based on how the player defines them: the more evil your beast, for

example, the more feared it becomes. *LittleBigPlanet*, an upcoming game from Media Molecule for Sony's PlayStation 3 game console, is built around player-made terrains and characters. But Wright's *Spore* is by far the boldest in terms of unleashing players' creativity. In *Spore* the players create life itself—starting with ooze-dwelling, one-celled creatures that learn, grow, and evolve into intelligent beings with advanced cultures and technologies, able to conquer their planets and outer space.

Computer gamers everywhere have eagerly awaited Wright's latest project since he began talking about it in 2000. *Spore* is finally due to be released this month, more than a year behind schedule. Wright attributes its recent delays to localization, the process of tailoring the game to different countries and languages. Others around the Maxis office cite the boss's high expectations. Wright concedes their point but shrugs it off.

"For games, it is a long time, but for me it's not a big deal," he says, sipping coffee in his cluttered corner office. "I'd rather spend a couple of extra years and have it be a big seller than short it by a year or two and have it be mediocre."

Spore is anything but. Other games may look and sound better, but few games are as original as this one. It offers players far more choice and open-ended play than any game before it. If *Spore* lives up to its creator's vision, it will likely be heralded as one of those milestones that redefines what a game can be—just as *Doom*, a first-person shooter game, pioneered fast-action multiplayer competition in 1993 and *Guitar Hero* delivered the thrill of music performance by introducing a guitar-shaped controller.

IMAGE: ELECTRONIC ARTS

SIM EVERYTHING: Through the course of the game, creatures evolve from single-celled organisms into hyperadvanced societies.

The anticipation—and pressure—is high. "I call Wright a genius because he truly is one of the most innovative developers out there," writes one gaming blogger. "*Spore*...is creating an entirely new genre."

The game unfolds through five stages, each of which riffs on an established genre of play. It starts, fittingly, in a two-dimensional world, with a single-celled organism that gobbles up microbes and plants to accrue DNA points. Once the spore collects enough DNA, an editing palette pops up that lets you design the next evolutionary stage of the creature's body. Your creature is then thrown into a three-dimensional environment where it must dodge predators and find a partner with which to reproduce. By the third stage, your creature is fully evolved and you switch to controlling its entire tribe, as you would in real-time strategy games like Electronic Arts' *Command & Conquer*.

Next up is the civilization phase, in which you can assemble vehicles and buildings to bring your tribe's city to life, in the spirit of *SimCity* or *Civilization*. If you succeed in conquering your planet and avoiding an enemy takeover, you graduate to the fifth and final level: outer space. Here the object is to fight off invaders and take over other planets.

Developing a game in which the players create all the key parts—the characters, buildings, and vehicles—poses an obvious conundrum: "There's no content," says Maxis technology fellow Chris Hecker. "Initially, the problem was, well, what is [*Spore*] supposed to be?"

When you boot up most games for the first time, you're immediately immersed in an existing world, complete with a cast of characters who behave in predetermined ways. Perhaps the game has tree-lined streets or castles with dungeons and moats. Maybe colonies of dwarves and trolls populate those worlds, or maybe gangsters do. These objects are all encoded in the game's original software exactly as the developers envisioned and animated them.

In *Spore*, that model doesn't apply. Almost nothing exists until the player makes choices about each object's shape and texture. To enable that design process, the relatively small team of 20 artists and seven programmers created a palette of editing tools. They think of it as an "artist in a box" or,

as Maxis software engineer Colin Andrews puts it, "Mr. Potato Head on steroids." Maxis released a stripped-down version of the tool palette, known as the Creature Creator, on 16 June to build buzz for the game. Eight days later, early adopters had created more than a million creatures.

To understand why the Creature Creator is so compelling, consider its incredible flexibility. Say a player wants to make a building. *Spore* provides a menu of architectural elements to tinker with: windows, doors, that kind of thing. The player clicks and drags the pieces onto a base structure and can stretch or shrink them along several axes. From the game's perspective, each building's design is simply a list of instructions; when the player is finished tinkering, those instructions direct the game engine to generate an image of the building and place it within the *Spore* world. Simple enough.

Then there's the process of making a creature, which offers a whole other level of variety and complexity. For instance, each creature can have any number of features and appendages—eyes, mouths, legs, feet—which can be stretched and curled like clay into outlandish shapes. But that indeterminacy presents an unusual problem: how exactly does a game company write software that generates realistic movements for "an eight-legged, two-headed thing with four mouths and no neck?" Wright says. "We don't know what we're animating."

To convincingly evoke even the wackiest animal a player could design, the game code had to be able to apply the knowledge of a human animator, on the fly—the ability to understand body language and subtle facial expressions and then to encapsulate those qualities abstractly in software. Wright decided to build *Spore*'s real-time animation around a technique called procedural generation. The "procedure" in *Spore* is a set of algorithms that execute a player's designs, generating entirely new content in the midst of game play. Other game developers have used the technique for years in a limited way, but no game has ever relied on it so heavily to create highly customizable yet lifelike creatures in real time.

So the Maxis team had little to go on as they tried to figure out how to make their exotic beasts move. Wright, who builds BattleBots for fun and possesses a voracious intellect and curiosity, decided to hit the books. He began reading up on biomechanics—in particular, the physics and physiology of how animals move. "Depending on the leg length and how supple the spine is," he says, "you can get a characteristic oscillation of the [torso] of the creature over the ground."

IMAGE: ELECTRONIC ARTS

COMPOSING CREATURES: An editing palette gives Spore players free rein to create unique and curiously realistic characters.

To get a creature to walk or run convincingly, the software engineers encoded an overarching set of rules on how to generate movement. The animation algorithms start by looking at the number of legs, the length of each leg, and the creature's bodily symmetry to calculate something called a walking ratio. If one side has twice as many legs as the other, for example, the ratio would be one to two. The algorithms will also compute the rhythm of a creature's footfalls—the length of time between, say, a front leg and a back leg hitting the ground in a single stride. The overall gait takes all these factors into account, along with the dimensions of the torso and head (or heads, as the case may be). The result is a convincingly lifelike motion.

The Maxis team then had to see if those movement rules worked on actual *Spore* beings. The team devised a huge menagerie of test creatures, observed how they stumbled around, and then

adjusted the software's algorithms, essentially creating a virtual island of Dr. Moreau. "We have these tests where we take 10 totally crazy, random creatures and run animation on them," Wright says. "And we find out that it works for these seven, but for these two the legs look weird and for this one the back isn't straight enough. We're refining those algorithms all the time."

To illustrate, Wright goes over to his Dell computer and, with a few pecks at the keyboard, brings up the game's Creature Creator on screen. He starts with a short, fat torso and attaches birdlike wings on its sides for ears. From a palette of eyeballs, he clicks and drags a pair of big round eyes and drops them onto the beast's shoulders. He continues to tweak the anatomy, equipping the creature with legs, arms, hands, and so on.

Even as Wright experiments with different looks, the beast begins to move—wiggling its newly attached limbs and blinking its new eyes. It even seems to show its approval of certain choices by smiling and nodding.

Based on each new creature's features and shape, the animation software determines the sounds it can make, the way it dances, and much more. A skinny beast with a beak and decorative tufts of hair may flutter its eyelashes and emit a high-pitched warble, while a hulking creature with spikes along its spine may blink slowly and communicate in a baritone growl. Those traits in turn end up influencing whether the creature greets other species with a friendly advance or with an attack, and the fate of its civilization depends on those nuances.

Wright clicks on a button to test the creature he's just designed. With short legs on one side and long legs on the other, the animal lumbers awkwardly but convincingly across the screen. Indeed, the little legs scurry just fast enough to keep up with the long ones. But Wright isn't done with him yet. "What would it take to make any creature sad?" he asks, tapping away at the keyboard. Suddenly this alien being adopts a recognizably sad pose, dropping its torso, curling down the edges of its mouth, and dully drooping its eyelids. You feel kind of sorry for the little guy.

This beast is relatively straightforward, but the Maxis team had to allow for the most twisted possibilities a player might dream up—for instance, a creature with no limbs. "Now the game has to deal with all the ramifications of that," says Hecker. "So how do you pick up a piece of fruit?"

In conventional animation systems, the concept of a limb may be encoded not as an object but rather as a set of spatial transformations that can be applied to a body. To accomplish this, an animator can assign labels to parts of a character's skeleton. When a character reaches for that fruit, the animation might state something like, "Rotate bone 1 from 0 to 52 degrees."

But in *Spore* the skeleton is unknown until the game is already in play. So instead of using labels, the programmers encode generic descriptions of each body part, referring to a specific limb by describing its context relative to other body parts. Let's say a creature throws a punch at a bad guy. The animation may dictate the action with instructions that would read something like, "Move upper leftmost grasper from rest position to a position parallel to your leftmost head, then move to some position relative to the enemy's topmost nose." The code analyzes the body in search of the parts that match that profile. In short, rather than directing bones to assume prescribed positions, animators are using higher-level directives to describe what the bones should do. This strategy was key to opening up a much wider field of character types and activities—though it certainly didn't make writing the game easy. The code can look for a limb by using a description that may be satisfied by one body part, several, or none. Coping with indeterminate results, while keeping the animation interesting by not simply ignoring extra limbs, drives up the complexity of the game code.

And then there are the creatures that are so weird they defy the game's generalized rules of movement. To catch those freakish cases, the code checks for certain features: for example, does the creature have any kind of paw or claw? If not, a separate set of instructions will govern the creature's movement, instructing it to use its mouth as a hand. And because the creature has no legs to use in calculating movement, it gets an inchwormlike slither. Rather than try to write very complex algorithms that cover every imaginable kind of beast, the programmers instead identified a few exceptional creature skeletons and wrote code that chooses different sets of rules for them. "It just kind of ripples out in a lot of different ways," Hecker says.

It's one thing to create and animate a creature in *Spore*, but Wright and his team knew that players would also want to share their creations. "What we saw with *The Sims* was that people loved downloading tools and creating stuff in the game," says Wright. Players routinely surfed Web sites specially set up for trading *Sims* add-ons, such as modifications to a character's appearance, houses, and furniture. But the experience was a hassle because players had to find and then import each item into the game one by one. "We wanted to basically make that [process] part of the game play," he says.

To do that, Maxis devised the Pollinator. This tool lets players easily search through the buildings, vehicles, and beasts created by other gamers and incorporate them into their own worlds. They can also sort through the stuff by theme—whimsical or *Wizard of Oz*, futuristic or "Futurama." This is what Wright means when he describes the game as "a massively single-player experience": it's a one-person game that can draw on many sources. While the *Spore* DVD will ship with some ready-made creatures and buildings—so that a player's creatures aren't initially running around in an empty universe—the rest of the content will come from the players, who can upload their creations to Maxis's game servers for others to access.

To store and sift through such a huge amount of data, the Maxis team had to compress the data files down to a manageable size. Here, the hurdle became "how do we keep the data rate really low so that even if I'm not on the Internet I can still have the local database with lots and lots of content?" Wright says.

With its detailed terrains and texture maps, one planet in *Spore* could occupy 10 megabytes of space on a player's hard drive. "We don't have the disc space to deliver a million planets," says Maxis art director Ocean Quigley. When a player creates a planet, an instruction list for generating that world is saved along with certain seed values, which are like keys that the software uses to reopen the world later. To conjure up lots of different planets for each game, the code requires that certain values or quantities be more or less randomly assigned. Algorithms embedded in the game's software can generate those strings of seemingly unrelated numbers, but the starting value—or seed—must vary so as to avoid generating the same string of numbers each time the algorithms are run. "We want the planets to be 'random,' but we also need to be able to re-create it exactly when you come back later," Andrews explains. "Storing the seed lets us do that."

The programmers also had to winnow down the list to just the core guidelines needed to reconstitute a planet—or building, creature, or spaceship. Sometimes that meant making tough choices that in effect curtail a player's creativity. Originally, for instance, *Spore*'s Creature Creator allowed players to design animals with looped spines. Unfortunately, doughnut-shaped animals raised all sorts of exceptions to the animation rules. The solution: bye-bye, doughnuts.

To manage the flow of so many player-created creatures and items, and to help players find content they like, *Spore* uses the same kind of collaborative filtering that sites like Amazon and Netflix have made popular, based on the preferences of other players who have chosen a certain design. Players will also be able to subscribe to *Spore*casts—a kind of RSS feed of content other people create for the game. As *Spore* spreads, stars of content design will likely emerge, as they have in *Second Life* and in other online gaming communities.

"I can imagine so many cool possibilities that we're just scratching the surface of," Wright says. He envisions *Spore* races centered around user-designed vehicles and flying games featuring users' spaceships.

But Wright's imagination stretches only so far. He anticipates the day when *Spore* players take charge and steer the game into unseen territory. "That's when the fans become an even larger designer [than us]," he says. "In some sense, we're kind of codesigning *Spore*. Fans are going to drive its future."