Editor: Mike Potel

Premo: DreamWorks Animation's New Approach to Animation

Paul Carmen DiLorenzo
DreamWorks Animation

Creating an animated film takes several years, hundreds of highly skilled artists across many different departments (see the "CG Animated Film Production Process" sidebar), and advanced software tools. At DreamWorks Animation, our artists use a mixture of third-party and internally developed tools supported by a high-performance computing infrastructure to digitally create our animated films. We continually update our tools and take advantage of the latest hardware developments to meet our filmmakers' demands to produce more captivating characters and stunning visuals with each new movie.

In recent years, those hardware developments presented a new challenge. In the past, computer performance roughly doubled every two years, following Moore's law, which states that the number of transistors doubles every two years on a CPU. However, in the last decade, we have seen these additional transistors serve more to increase the number of CPU cores than increase the speed of individual cores. For the software industry as a whole, this has led to a crisis, and for the high-performance software we use, the problem is particularly acute because we rely on that increased core speed to deliver faster software to our artists.

To respond to this paradigm shift and scale with new hardware, we rearchitected our proprietary animation platform to take advantage of multiple-core computing. Premo is our new animation tool engineered by DreamWorks Animation technologists to enable artists to manipulate character performances in our movies.[1] Premo uses Apollo (www.dreamworksanimation.com/apollo), our new cloud-based digital design and CG media platform used for the creation and delivery of images, video, and other forms of media.

Premo was first used on the feature-length animated film *How To Train Your Dragon 2*. Since then, DreamWorks has made a full transition to Premo, and it has been used on the films *Home* and *Penguins of Madagascar* and is now being used on five films currently in production. Premo has exceeded our artists' expectations in both performance and usability and has resulted in greatly improving animator productivity and satisfaction.

Premo has not only resulted in a better user experience but also some key returns on investment for DreamWorks Animation. The training time for a new animator on Premo has been reduced to one to two weeks, compared with the 13-week course for the previous software. This enables animators to work on a movie soon after they are hired, whereas before we had to wait for them to be comfortable with proprietary tools. Initial studio estimates project up to a 20 percent reduction in production costs for movies using Apollo, starting with movies released in 2015. In addition, we feel that Premo offers audiences the highest quality entertainment experience with characters that provide the fullest range of visual emotion and expression.

Inspiration and Design

When designing Premo, we were able to leverage the many man-years of experience from our animators, allowing us to develop an application specifically suited to animation. To take advantage of this experience, we kicked off Deep Dive groups to focus on the main areas of the application. These groups discussed a specific topic for a few weeks and generated a substantial number of ideas. These ideas were distilled down into a set of Guiding Design Principles that outline key project requirements:

 Published by the IEEE Computer Society

CG Animated Film Production Process

CG animated films are more complex than most people think. Each generally takes about five years to develop as a result of the planning and production of all the layers and assets that go into making a film. Great films begin with great concepts. Some of our ideas are completely original, whereas others are inspired by a variety of sources, including children's books and comic strips. Once we've settled on an idea, the first step is to write a script.

Storyboards

Once a script is ready, storyboard artists will imagine how the words will translate into actions and pictures by drawing a series of sketches to tell the story. The drawings are digitally strung together to create a story reel. (Imagine a flipbook that lets you see how the drawings flow together.) We combine that with temporary music, sound effects, and dialogue and work with the movie in this form for about 18 months.

Visual Development

Once the story reel is underway, our visual development department begins to plan the look of the film, developing the style, tone, color and overall artistic approach to each and every sequence. Everything has to be designed, from the major characters to the smallest of props. Thousands of drawings, paintings, blueprints, sculptures, and models later, our development artists have designed a fantasy world and characters to tell the story.

Modeling and Rigging

Modeling artists digitally sculpt the characters and environments in our films by collaborating with the art department to realize design concepts as tangible 3D forms. The modelers start with this wire frame sculpture that breaks down the design into workable geometry. A character technical director (CTD), also known as a *rigger*, will determine how the character must move and where the bones, muscle, and fat would be under their skin. Joints and various mathematical operations are then employed throughout the character's body, face, hair, and clothing to make it bend and deform like a living creature. Finally, rigging works with animation to design and build an extensive set of controls for the character so animation can pose every part of this digital puppet and bring it to life.

Layout

Layout artists interpret and recreate the hand-drawn 2D storyboard panels in a 3D CG environment. In doing so, these artists determine the 3D camera placement and motion. Working with rough versions of the characters, lighting, effects, and environments, artists create the cinematography for the film.

Character Animation

Once the sequence is working well in layout, the animators start bringing the characters to life in the computer. They articulate the thousands of controls that were created during the character-rigging phase to bring each character to life and to synchronize them to the voice performances.

Surfacing

Coming out of modeling, characters, props, and environments are flat and grey. The surfacing artists add the colors and textures to these elements, making surfaces look smooth and shiny like glass, bumpy and gritty like dirt, and fuzzy and soft like wool.

Effects

In a live-action film, it's easy to photograph things like leaves blowing in the wind, waves at the beach, or even footprints in the sand. In computer animation, these simple things are all designed and animated by the effects artists. In other words, if it's not acting, but it moves, it's an effect.

Lighting

Lighting artists utilize the computer to "paint" with light, bringing the final color, look, and illumination to the film. Lighting is the first time we get to see animation, surfaces, grass, trees, water, crowds, and effects all working together. Lighting does this by creating illumination for the scene. It creates the mood and atmosphere to support the story. Lighting leads the viewers' eye to the critical elements of the frame so that the audience is looking exactly when and where the filmmakers want them to look.

- direct control to "reach in and grab" the character;
- fast refresh of characters and environments in real time;
- representative high-resolution, fully deforming characters and environments, indicative of the final movie;
- creative workflows that allow animators to pose, draw, and explore in a more natural way;
- fluid and seamless flow between different tasks such as posing and adjusting motion across time;
- flexible and fully configurable interfaces that allow multiple views of animation controls;
- intuitive browsing, previewing, and editing of the entire movie segmented into "shots"; and
- immersive experience that keeps the animator "in the moment" with their creative thoughts and ideas.

To create a clear and consistent vision for the project, we borrowed the storyboarding approach

Figure 1. Premo digital pen interface. An artist is using a Wacom Cintiq and a digital pen in Premo to edit Hiccup, the main protagonist character of the *How to Train Your Dragon* franchise.

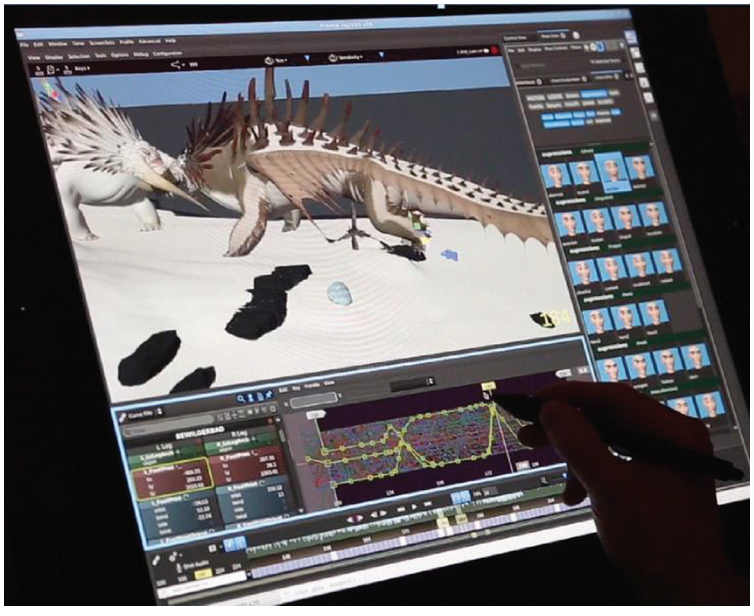

Figure 2. Premo character complexity. The model for the Bewilderbeast, a new dragon character introduced in *How to Train Your Dragon 2*, is 20 times the size of Toothless, the featured dragon from the original *How to Train Your Dragon* movie.

that our filmmakers use by creating mockups. Many different mockup types were used, such as images, movies, and Flash animations. These mockups helped demonstrate the interaction and specific workflows desired by animators inside of Premo.

Transition to Premo

Prior to Premo, animators used Emo, DreamWorks Animation's previous generation proprietary software, which was used to animate character performances in more than 25 of the studio's animated features. When developed, it was a state-of-the-art, award-winning tool; however, its architecture did not adequately support the multicore computing model that was needed to meet the interactivity goals of the project.

With Emo, changes made to character movements were typed into a spreadsheet-like interface, and it required significant time to calculate and render the results on screen. To help speed up this compute process, low-resolution versions of the characters were used, and environmental detail was removed. However, this meant animators could no longer experience their work as it would appear in the final movie, forcing them to make many guesses in animation performance, which often later required additional fixes. This method was not only time-consuming, but it also limited the number of iterations and broke the stream of creativity.

Premo's rearchitecture, based on modern multicore computing, has resulted in greatly enhanced real-time performance and interactivity. As a result, with the transition to using Premo, the animators' workflows and experiences radically changed, providing a variety of new capabilities and functionality. To fully capitalize on the Premo rearchitecture, we deployed to each animator a HP z820 workstation that uses Intel IvyBridge with 10 cores at 3 GHz, 96 Gbytes of RAM, Nvidia K5000 video cards with 4 Gbytes of RAM, and a 960-Gbyte solid-state drive (SSD).

Animator Workflows and Experience

The greatest experience shift for animators is to no longer need to wait for the recalculation of their scene to see the result of edits. They can quickly iterate on their animation, try different ideas, add more subtle motion and expressions, and ultimately realize their vision for the character performance.

This significant speed improvement enabled new workflows to be introduced. For example, animators can now work interactively with a pen on a pressure-sensitive tablet. They can precisely control a characters' face and body by simply moving their pen to the exact degree desired and receive real-time feedback (see Figure 1).

Animators desire the ability to work with the highest quality representation of the characters and environments. With Premo, animators manipulate high-resolution, fully deforming characters in their scene. They can load many characters and detailed environments to judge the character's performance in the context of the scene. Figure 2 shows a loaded scene in Premo where two Bewilderbeast dragons are battling. The Bewilderbeast dragons are large, detailed characters, yet Premo is able to stay interactive even with this level of complexity.

Advanced Animation Features

The guiding principle of creativity was a high priority for the animators. Toward that goal, we added a fast, high-quality drawing system into Premo that provides the ability to sketch in a shot using a digital pen (see Figure 1). Many of our highly trained 2D animators, whose natural language includes being able to draw by hand, use the drawing tools to develop ideas for their work. Animation supervisors use these drawings to provide notes to the animation team. This type of collaboration has proven to be faster in communicating character performances.

To help animators animate across time, we added a motion path and ghosting feature into Premo (see Figure 3). These features allow the animators to see the character in past and future frames, enabling them to effectively work across multiple frames in a single view. The motion paths provide a simple curve visualization and editing capability for a single animation control, and the ghosting feature provides a full character representation over time.

Continuity of character performances across multiple shots is important to maintaining visual consistency. Previously, our animators were confined to working within a single shot and were required to close and reload the application to work on other shots. To evaluate continuity between shots, they would have to generate movies for each of the shots, which was tedious and time-consuming. In Premo, we created the ability to open an entire sequence, choose the shots to be worked on, and add them to the session (see Figure 4). The animators can then make edits on any of the active shots and see how those changes play with the surrounding shots in the sequence. Because Premo is connected to Apollo's cloud platform, it can access and play videos created by other departments in the production pipeline. For example, they can access the latest renders from the lighting department of the shot they are currently working on to easily compare versions.

Figure 3. Premo motion path and ghosting feature. With motion paths (purple line) and ghosts of the character over time, the animator can visually and quickly set the precise position and timing they want for the character's motion.

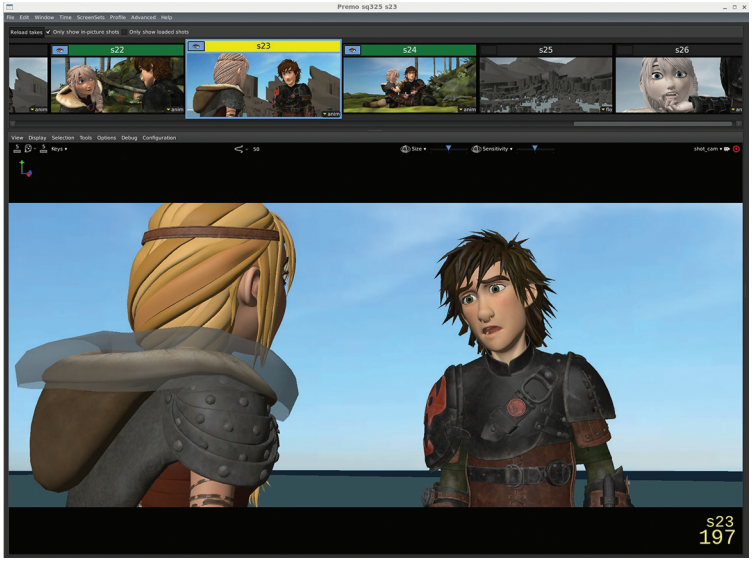

Figure 4. Premo Shot Browser. Using the Shot Browser viewer at the top of the Premo interface, animators can quickly switch between shots, choose the department view (such as layout or lighting) they want to access, hover over a shot to see a movie clip, or choose a set of shots to play continuously in the Scene View (main view) to ensure consistency in their animation.

Architecture and Technology

Three architectural goals helped us achieve the features and animator experience of Premo. The first is to maximize utilization of computational resources by distributing heavy computations across local cores and continue to be highly scalable as hardware capabilities increase.

Even with this highly scalable architecture, the application may not be able to maintain interactive rates given a sufficiently heavy workload. Therefore, the second goal is to build an architecture that treats computation as a service. This service must completely decouple evaluation from the client to ensure that the application is consistently responsive to users.

By separating the computation from the client, we are no longer restricted to doing computation on the local cores. This enables the third goal, which is the ability to allocate more cores to the artist when required. Because we expect the complexity of characters and environments to continually increase in order to achieve our filmmakers' goals, we need to be able to bring additional computing

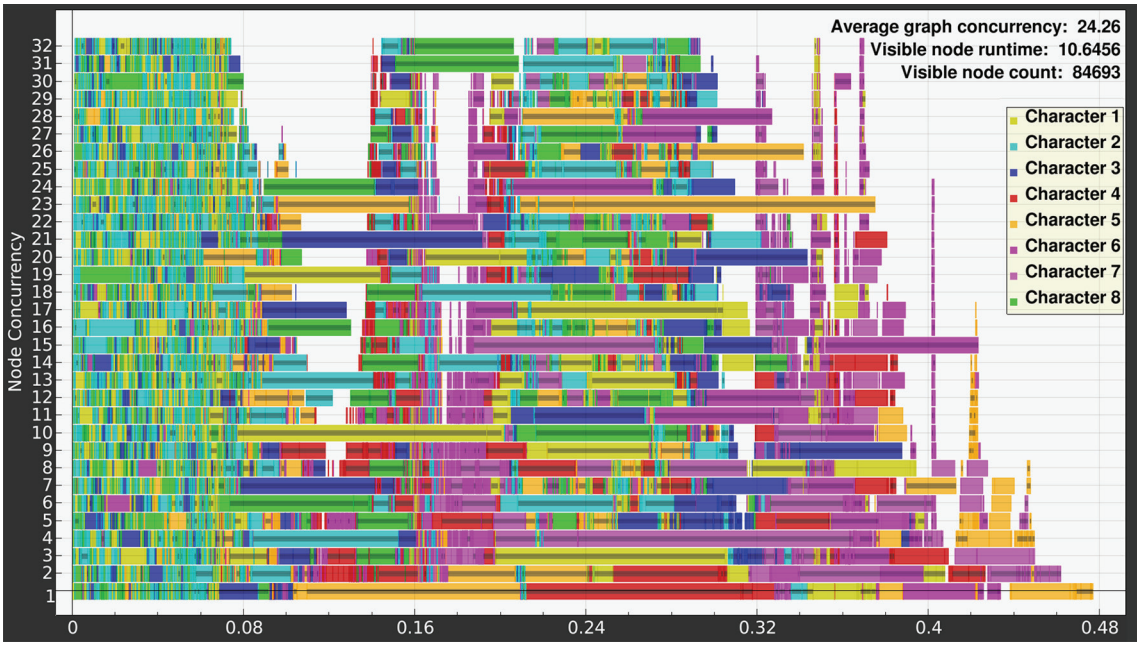

Figure 5. LibEE visualization tool. LibEE shows the execution of each node and stacks them when they are executed at the same time to show the degree of parallelism. Colored areas mean that the cores are being used, and the white space shows inactivity. This example depicts a single evaluation frame of eight characters on a 32-core machine. During the evaluation, there are times when all 32 cores are busy, indicating that performance is limited by the hardware resources, rather than the rig itself or an evaluation engine. Because future machines will almost certainly have more cores, this means we are well placed to continue to scale performance of our engine with future hardware.

resources to bear if the local animator workstation is insufficient.

These goals ensure that Premo can achieve the Guiding Design Principles even as workload complexity increases over time.

LibEE

DreamWorks Animation partnered with Intel to develop LibEE,[2] a high-performance, multithreaded dependency graph evaluation engine inside of Premo. LibEE is designed to evaluate the characters in a shot as efficiently as possible.

Before animators can create character performances, our character rigging department builds the skeletal structure of a digital puppet, called a *rig*. The rig defines the available controls on the character that the animator can use as well as the degree of motion that can be applied to those controls. Some of our characters have as many as 5,000 controls, and a rig can contain up to 150,000 individual *nodes*, each of which is an element of computation that performs one particular action within the rig. All these nodes are chained together into what is known as a *dependency graph*.

LibEE uses Intel's Threading Building Blocks library (http://threadingbuildingblocks.org) to achieve fast, scalable evaluation. The primary benefit of LibEE over other dependency graph evaluation engines is its ability to evaluate multiple parts of the character in parallel (such as arms, legs, and other independent parts of the character), providing a significant speedup over more traditional nonthreaded evaluation systems, up to 10 times in some cases.

One important concept is that in order for our rigs to evaluate in parallel, not only does the evaluation engine need to be multithreaded, but the rigs themselves also need to be structured to allow graph parallelism. As a simple example, if a rig contained a set of nodes that each attached one toe to a foot, and those nodes were connected in a series that each depended on the one before, then the evaluation engine would have to evaluate each toe at a time. However, when the nodes are connected in parallel, so there is no dependency between individual nodes, then all the toes can be evaluated at the same time.

Building a rig that allows the engine to extract maximum scalability from the workload was a new challenge for our production artists.[3] To help meet this challenge, we implemented profiling tools that allow riggers to visualize the data flow and node evaluation in a graph (see Figure 5). Artists can identify which character components are running in parallel, determine occurrences of graph serialization bottlenecks, and discover where unexpected dependencies may exist between parts of the rig. The tool highlights nodes on the critical path (along

Figure 6. Premonition system process. (a) When an animator edits a character, that prompts Premonition to begin recomputing the surrounding animation frames, indicated by the orange bar. (b) The brown bar clears as those adjacent frames compute. (c) The animator can then begin playback to watch the animation, which can display in real time because Premonition has precomputed these results. At this time, Premonition will continue to process the remaining frames in the background. (d) Finally, the animator can change time manually, and Premonition will adapt and predict new frames to compute based on the new time.

the bottom horizontal bar), which determine the overall best possible runtime of the graph.

Premonition

Premonition is one of the key systems inside of Premo. Its purpose is to take advantage of the idle time (sometimes fractions of a second) between the edits an animator will make on a character. While animators are posing on a single frame, they are more interested in seeing how that edit affects the motion of the character over time. Therefore, after an animator makes an edit, Premonition will start to compute adjacent frames immediately (see Figures 6a and 6b). By taking advantage of this animator workflow, we maximize utilization of the computational resources on the workstation.

Because we tuned our architecture toward animator workflows, Premo allows the animator to make an edit and playback the shot while Premonition processes the shot in the background (see Figure 6c). Premonition also has the ability to adapt and predict (see Figure 6d). As the user changes to a different time in the shot, Premonition will adapt and adjust the frames it computes based on the new time the user wants to view. From there, it will try to predict and compute the frames the user will view next.

High-Performance Evaluation Architecture

Premo's high-performance evaluation architecture uses a layered architecture with LibEE at the bottom and the client layer on top. The client layer contains the UI providing user input in the form of animation curves and output in the form of geometry to display on screen. Between LibEE and the client (the middle layer) is the Graph System, which acts as the application's interface into the graph.

To treat computation as a service, the Graph System manages evaluation requests through a subscription model, where the client subscribes to a

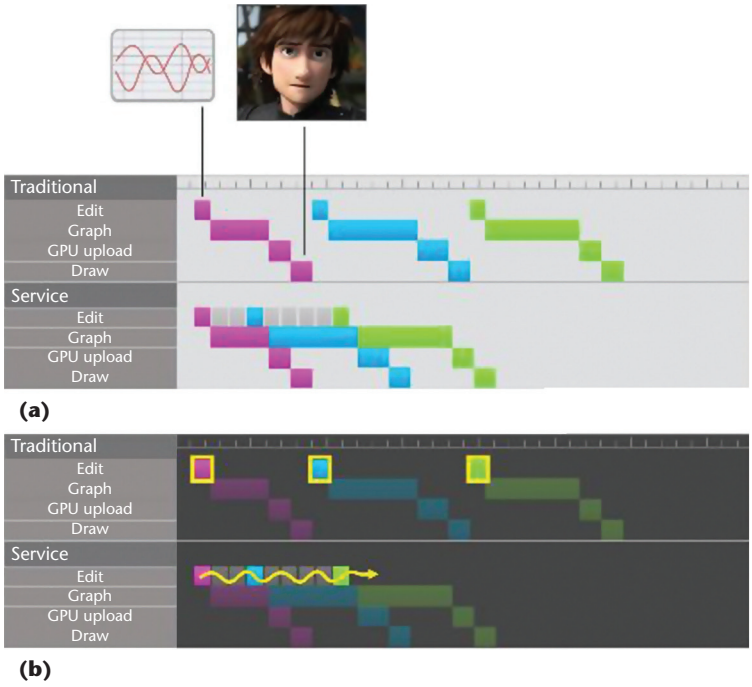

Figure 7. Treating computation as a service in Premo. (a) The traditional approach couples the application with the graph. This results in the application waiting for the graph results and under-utilization of the graph. In comparison, our service-based approach allows the application to stay responsive and fully utilizes the graph. (b) With the traditional approach, the user would receive results only when an edit is fully realized by the system. This leads to a choppy experience for the animators. In contrast, our service-based approach produces a continuous, analog experience for the animators. (Courtesy of DreamWorks Animation.)

particular output of interest from the Graph System over specific time frames. The Graph System uses these subscriptions to orchestrate graph evaluation with LibEE and then batches these requests for optimal parallelism. When LibEE finishes an evaluation, the Graph System pushes the results back to the client to immediately display the results.

To illustrate the unique benefits of treating computation as a service, let us step through a single cycle of an application utilizing the traditional approach (see the top row of Figure 7a). This approach begins when a user edits animation curves. This kicks off a graph evaluation to process the curves and compute new character geometry. Finally, the geometry uploads to the workstation's GPU to display the results on the screen.

In the traditional approach, graph evaluation is coupled with the application event loop where the main UI thread is constantly waiting for the graph to finish before it can update. In our service-based approach (see the bottom row of Figure 7a), we have essentially compressed all the colored blocks to the left, creating a full pipeline of edits, evaluation, and display. By decoupling each of these tasks, user edits can be processed at a higher rate than graph evaluation, which is represented by the gray boxes. Furthermore, computing resources are now fully utilized by continuously keeping the graph busy, resulting in an increase in overall throughput of our pipeline.

The outcome of this approach for the animator experience is significant. As seen in Figure 7b, the service architecture inside Premo returns 3D animation back to a continuous and analog experience, not a discrete one as experienced in traditional 3D editing applications. Creative processes, such as drawing on paper, are inherently analog, smooth, and fluid. When the industry moved to computer animation, the creative process became discrete due to the nature of engineering. However, with Premo eliminating the wait time on the graph, animators are able to return to an analog experience, and the result is a natural interface for 3D animation.

Solving Advanced Animation Challenges

Character constraints are created when an animator wants one character to drive the motion of another. For example, if the character Hiccup is riding his dragon Toothless, an animator can constrain Hiccup to Toothless in Premo. When the animator moves Toothless, Premo will automatically compute the new position of Hiccup. A more complex example would be characters locking arms and influencing each other's motion. The challenge is that character constraints often introduce infinite evaluation loop cycles in our graph. Furthermore, when adding numerous constraints, it is difficult to maintain interactive rates. Therefore, we build in native support for handling cyclic dependencies. In the Graph System, we manage constraints and track the resulting cyclic dependencies, while leaving LibEE as a directed acyclic graph (DAG). To evaluate the cyclic dependencies, we do a multiple-pass planning stage based on constraint priorities. The resulting plan is then executed by LibEE.

The second animation challenge is the seamless integration of editing and simulation. Character simulations (such as hair and clothing) increase the visual fidelity we give to the animators. We want the animator to be able to see live simulation results for the frame they are viewing when they are posing a character. When an animator manipulates a character that has simulations, the simulation requires the evaluation of previous frames before the current frame can be displayed. This requires extensive computational resources. In contrast, when an animator manipulates a character without simulations, the dependency on

previous frames does not exist and the application only needs to evaluate the current frame. The challenge is to ensure the architecture is aware of these temporal requirements and can do the necessary computation to achieve the desired result for the animator. To solve this challenge, we introduced the concept of time-based dependencies into the architecture. Simulations are required to describe their time-based dependencies to ensure that the current frame being displayed to the animator is always up to date.

From the onset of the project, we set out to build a tool for animators, inspired by animators, with the motto of "no compromises." The result has been the achievement of both the design and architectural goals we established for ourselves. We know our tools will continue to make effective use of future hardware, enabling us to keep up with our demanding production environment.

Acknowledgments

I thank Martin Watt, Matthew Gong, Fredrik Nilsson, Anna Lee, and Bill Ballew for their help and guidance with this article. I also thank all the engineers, artists, and managers who worked on this amazing software (past and present) and for creating significant IP in this area as well as the studio leadership for their support of this program. Special thanks also to Theresa-Marie Rhyne for encouraging the development of this article.

References

1. M. Gong et al., "Premo: A Natural-Interaction Animation Platform," *ACM SIGGRAPH 2014 Talks*, 2014, article no. 3.
2. M. Watt et al., "LibEE: A Multithreaded Dependency Graph for Character Animation," *Proc. Digital Production Symp.* (DigiPro), 2012, pp. 59–66.
3. G. Zimmermann, K. Ochs, and R. Helms, "Building Highly Parallel Character Rigs," *ACM SIGGRAPH 2014 Talks*, 2014, article no. 27.

Paul Carmen DiLorenzo *is the R&D project manager for the Animation and Rigging Program at DreamWorks Animation. Contact him at paul.dilorenzo@dreamworks.com.*

Contact department editor Mike Potel at potel@wildcrest.com.