Earthquake! Building a Pipeline to Destroy Los Angeles in *2012*

Haarm–Pieter Duiker* Digital Domain
Osiris Pérez† Digital Domain
Masuo Suzuki‡ Digital Domain
Rito Trevino§ Digital Domain

2012 presented a set of challenges that were unique in scale, even for an experienced company like Digital Domain. The LAP sequence follows the progress of a plane flying through Los Angeles as it is destroyed by a massive earthquake. From this aerial perspective, some shots had as many as 6000 individual objects. Each needed to be placed, animated, run through effects simulations, and finally lit and rendered. The time constraints and scale of this project meant that most of those steps would happen in parallel, and assets would be constantly modified, published and updated by each department as shots progressed.

1 Organising Chaos

Bento is the name of the system designed to meet the core challenge of allowing artists working with large numbers of assets in multiple departments and packages . It introduced a new data structure for specifying object membership, components and placement in the scene. It was implemented as a core Python library and Python plug-ins for Maya and Houdini. Bento also built on and extended an existing set of tools for working with the Houdini-native bgeo geometry data format for interchange between packages.

Bento's two main data types were the Box and the Asset. A Bento Box is a hierarchical structure contains Assets and other Boxes. It is the primary means of representing object membership and placement. A Bento Asset is an assembly of all the components associated with an object in the asset database. Geometry, rigs, shaders, textures, animation curves and baked deformation are all examples of components that were stored as separate entries in the database and that could be updated and published individually. The Bento Asset could be reconstructed automatically from its components for the purposes of editing or rendering. The Bento Asset and Bento Box XML representation could be translated back and forth from scene object hierarchies in Maya and Houdini, complete with transforms and animation data.

Translating scene hierarchies into the Bento Box XML representation enabled the Layout team to publish a lightweight file that listed the appropriate assets with initial transforms for a shot. That Box could then be loaded independently into Maya or Houdini where the Animators, Effects Artists and Lighters could add their changes without conflicts. After each department or artist had their work approved, a new version of the Asset and the Box that contained it would be merged with previous versions and published for everyone else to use. Furthermore, a robust set of updating tools allowed artists to view what changes had been published by other departments since they last loaded data from the database and to either selectively or in total update the Assets in their scene with those changes.

Interactively, the Bento system in Maya allowed artists to manipulate very large shots by giving the user the ability to control the level of detail used to display each Asset. The initial representation of each Asset was very lightweight, a locator. From there, the user could change any or all of the Assets to display as bounding boxes, static models, live rigs or geometry caches. Without the ability to switch resolutions for individual Assets, it would not have been possible to load and manipulate our largest scenes in memory.

In a similar fashion, Houdini artists could load the same Box XML description and manipulate Assets as well as create additional ones Assets to hold new effects like dust, debris, and simulations.

2 Rendering All That Destruction

As mentioned earlier, the Houdini native bgeo format was our standard for geometry interchange. A set of tools existed for reading and writing bgeo data from Maya. These were further extended to handle some of the new requirements presented by the show. One issue that hadn't been fully tackled yet at Digital Domain was rendering bgeo data directly in packages like Renderman and mental ray. A custom plug-in named geotor was developed to address this problem.

A typical shot, containing hundreds of assets

geotor's most core functionality was dynamically translating bgeo geometry data into rib stream object data. It went above and beyond that core requirement by adding support for specifying textures, shaders, motion vectors, primitive variables and user attributes using external XML formats published by the Texture, Lookdev and Lighting teams. The motion vector support allowed the renders to maintain proper motion blur even in the face of frame-by-frame topology changes.

All the functionality of geotor combined nicely with Bento's knowledge of the published geometry, texture, shader and lookdev components for a given Asset. Using these two systems together, Lighters in particular were able to manage, and render scenes with an average number of assets in the high hundreds, with individual scenes going well into thousands of assets. Wrangling data coming from all of the upstream departments and getting it all to render was no small task on a show like this one. Without Bento and geotor, it would not have been possible.

Acknowledgements

We would like to acknowledge the following persons for their valuable contributions to this project: Phil Peterson, Remy Torre and Mike Meckler.

*e-mail: hpd@d2.com
†e-mail: osiris@d2.com
‡e-mail: masuo@d2.com
§e-mail: rito@d2.com