Virtools™

Game Prototyping in Dev

WHITEPAPER

by Bertrand DUPLAT

President and CTO of Virtools

and

François DUJARDIN

Virtools Expert

What is a prototype and what is prototyping ?

Game prototypes come in various shapes and sizes, as varied as the reasons that initiated them; however, there is a common denominator on what we call game prototyping.
Prototypes are not just animated films giving a taste of both art and gameplay- its atmosphere, narratives - or even a non-interactive taste of gameplay. A game prototype has to be playable, even if it is far from the full finished game - just a make-believe level or two, or even an isolated aspect of the game. For instance, you may want to preview car controls and associated camera movements in a car racing game or the proper integration of models and animation with new animation algorithms.

In the playability lies the primary motivation behind creating the prototype, because a game prototype is in fact a tool to deal early on with interactivity and gameplay.

Here are some typical production issues that Virtools users answer through prototyping:

- Technical decisions on how to model a car racing track: prepared track tiles versus path lofting. Prototyping validates the feasibility of both methods, highlighting their pros and cons, and assesses the workability for a graphic artist in an actual production setting.
- Technical decisions on how to build a character skeleton, and specifically how to position its root bone, knowing the types of actions he has to accomplish. Prototyping helps understanding the pros and cons of positioning the root bone between the feet, versus on the pelvis. It also helps decide how to orientate it during moves like wall sliding, etc… The final decision is to choose a solution that's compatible with both the programmer's and artist's toolsets and habits.
- Camera movements. Since camera positioning and behavior are key elements to the playability and feel of any game, prototyping allows trying out different approaches for the camera. For instance, in sports games, a correct camera positioning is crucial when the player has to aim, and prototyping helps find the right position, so that the player and potential target are in view at the same time, at an angle that lets the player aim precisely.

Why Prototype Games ?

A game prototype is a tool made and used in the early process of designing an final complete game. Based on our long-standing experience with game developers using Virtools Dev, we found three primary reasons for building a game prototype:

- As a communication/sale tool
- As an exploration tool
- As a pre-production tool

Prototypes as a communication/sale tool :

- **To find a publisher**. One obvious goal for a game development house or designer with a concept is to make the prototype, on top of the game design documents, in order to convince a publisher to finance the project. And the prototype matters even more if it demonstrates the gameplay's quality or originality.
- **To convey a clear in-house vision of the game**. Here the goal can be two-fold: on one hand to convey internally the quality of the concepts and design to the in-house producer and get the game design going; on the other hand, to carry a vision to an overall game development team (artists, musicians, game designers, programmers, project managers...) to stimulate them and get them all going in the same direction.

Prototypes as an exploration tool :

- **To validate gameplay ideas**. New gameplay ideas need to get tested, refined and validated before being successfully added to a game. What if the ideas seem good and fun on paper but are tedious, bring nothing, or are just not understandable by the player? And, if these ideas are good, how to get them right, to get maximal benefits and effects from them.
- **To discover a whole new gameplay** . Even more so than new gameplay ideas, creating a whole new gameplay concept, whether at the frontier of an existing genre or even outside of it, is a highly difficult and perilous exercise. In that case, a playable prototype is an invaluable and unavoidable tool for building the gameplay to satisfying maturity (or deciding to abandon it).

Prototypes as a pre-production tool :

- **To solve technical issues**. Ideally during pre-production phase, all foreseeable technical and "artechnical" uncertainties and difficulties must be resolved. This must be done in order to anticipate and remove all future production blocks - or, worse, U-turns - and open the way for a free flowing, streamlined production process. Here prototyping is used to try out prospective techniques, whatever they might be: animation engine issues, AI or any other technique that need to be validated before production starts.
- **To dimension production in terms of media, code and their integration**. Pre-production is also when the final production process is put together, including team sizing, production planning, and integration techniques… Determining the scope is key to get the game in time and on budget. In order to estimate it accurately, prototype-based tests are key to make sure all future media (images, 3D models, animations, sounds and music) will work well with the code, and to make sure the media integration techniques work.

Nothing is more disastrous than finding out, once all that the media does not fit properly with the engine or gameplay context and that they have to be all redone…

Virtools SA
93, rue Vieille du Temple
75003 Paris - FRANCE
Ph: (+33) 1 42 71 46 86 / Fax: (+33) 1 42 71 86 53

www.virtools.com

Virtools Canada Inc.
1250, Bvd Rene Levesque Ouest/Suite 2200
Montreal QC H3B 4W8 - CANADA
Ph: (+1) 514 989 3126 / Fax : (+1) 514 989 3127

These diverse motives for building a prototype also show where prototyping fits at different stages of the overall game making process: early on, at the concept stage and up to pre-production.

The prototype's role as a tool is to attract the bright side - find financing, convey vision, create and refine gameplay - as well as stave off the infamous dark sides of game development: avoid designs that show their weaknesses too late, avoid technical roadblocks or catastrophic production planning mistakes.

How to prototype in Dev: Techniques and Methodologies

Team and Timing

A typical prototyping team can work around three people: one artist for the 3D models, texture and animations; a game designer; and a Virtools Dev Scripter to create the interactivity. In certain cases, the Game Designer can also be either the artist or the Virtools Dev Scripter. Prototypes rarely require C/C++ programming.

A prototype can be created in as little as 3-4 weeks, with initial playability already in place usually after one week. Achieving the earliest playability, in a week or less, should be a target goal.

Questions to be answered

Before starting the prototype, its purpose should be well defined. In which of the previous categories (communication/sale tool, exploration tool, a pre-production tool) does it fit? What questions need to be answered through this prototype?

If the prototype serves as pre-production tool, the production process of the prototype matters as much as the final prototype itself, because some of its design (and implementation) aspects will be reusable in the final game. Here the pre-production prototyping process itself should be a mock-up of the final process. However, if the prototype's final goal is as an exploration tool, keeping an open mind is more important, since too much planning might kill chances of making unforeseen discoveries. Therefore, prototype production techniques will differ depending on the final goal.

Depending on the prototype's purposes, what should be its core gameplay functionalities? Large multi-purpose prototypes with numerous functionalities can best be built and implemented by classifying them according to their desirability: "must have", "should have", and "nice to have". This allows the team to work iteratively: implement essential «must have» features first, then develop the full prototype in phases. Once «must have» functionalities have been integrated, the prototype is quickly functional in its basic form. The prototype can already be used to answer important questions without wasting any time.

A modular architecture

The primary hurdle in creating a prototype is allowing collaborative modifications to occur as quickly and as often as possible to continuously improve the prototype both in terms of art and gameplay. This iterative process among all team member requires adapting the prototype's architecture accordingly.

Gameplay and interactivity should be broken down as modules (e.g. player moves, camera moves, UI...). These modules should be as independent as possible, with inter-dependencies clearly highlighted and isolated, to allow the switching off or replacement of modules. Since this is the time to specify interfaces and constraints between the various bricks/modules of the game - later to be validated in the prototype - this is when to define production team's working process. This is where prototype production process helps formulate the final game production process!

When using Virtools Dev with a modular architecture, the prototype's core is a 'shell' .cmo file containing only the core script that dynamically loads all the modules as .nmo files, using external .txt or XML-like parameter tuning files. These later files contain what can be modified by the game designer through a standard text editor, enabling him to tweak gameplay parameters autonomously. Parameters can also be modified inside Virtools Dev with a button/slider GUI to validate them on the fly. In the same way, media is kept external so that the artist can modify them and integrate them in the prototype with minimal interference or support from the Virtools developer.

The modules' interdependencies take the form of parameters (usually in the form of parameter arrays).This modular approach permits the playing of content even if some modules are missing or still at an early stage. Theses modules can therefore evolve separately. Different modules can even be developed in parallel by different people. When working on these various modules, testing them in the overall context is very easy since launching the prototype will dynamically load the latest versions of the various modules and link them together at play time.

In the final prototype, the entire content is merged as one file.

Media

The idea is to start with very simple shapes that are later replaced with final 3D models. e.g.a cube for a car, existing animated characters...) These models are iteratively replaced as new better versions become available. So testing interactivity and gameplay can begin even before the media is produced.

Of course, to facilitate media replacements (made easy in Virtools Dev 2.5 and later versions), the rule of thumb is to respect topology and hierarchy. More precisely, (e.g. for vehicles or mechanical parts) scale, pivot points, and axis of rotation

Virtools SA
93, rue Vieille du Temple
75003 Paris - FRANCE
Ph: (+33) 1 42 71 46 86 / Fax: (+33) 1 42 71 86 53

www.virtools.com

Virtools Canada Inc.
1250, Bvd Rene Levesque Ouest/Suite 2200
Montreal QC H3B 4W8 - CANADA
Ph: (+1) 514 989 3126 / Fax : (+1) 514 989 3127

for objects should be well placed even in the crude 3D forms. Bone hierarchy should be identical between early and later characters.
At the end of the media integration process - especially when the goal is to find a publisher - the final media should be rich, complex and amazing in the final prototype.

Code and behaviors

Here the rule of thumb is to avoid low level code (C/C++) as much as possible. Most of the time, C/C++ coding to create new behavior building blocks is largely unnecessary for prototyping; however, it can be useful if one of the prototype's goals is to test out a specific algorithm. In that case, C/C++ and VSL (Virtools Scripting Language) provides the two ways to access low level functions in Virtools Dev. Of course, existing proprietary code such as AI routines or special visual effects (used in the final game)can be easily packaged into proprietary behavior building blocks that can then be used in the prototype.

So a thorough knowledge of Virtools Dev's existing behaviors and their capabilities (collision, character animation, particles, visual effects, UI...) is very important. . Behavior building blocks from additional Virtools Packs, such as the Physics Pack, Multiuser Pack, AI Pack and VR Pack can be very useful too.

Capitalising on one's own scripts by saving them into behavior graphs and scripted objects, (i.e..nmo or .nms files) provides maximum efficiency when reusing them on subsequent productions.

Testing, Debugging and Tuning

The good news is that a prototype's testing and debugging phase is not as critical and time-consuming as it is for a complete game. Often times, bugs can easily be circumvented since the goal again is often to validate the gameplay or a gameplay aspect. Bugs should however be listed carefully, since they may provide hints on the kind of problems that may be expected during actual production.

Versioning

While using a version control system during prototyping is not absolutely necessary, it ensures that collaborative work between the artist, designer and developer is seamless. Since the prototype is split in files modified by the scripter (.NMO and .CMO files), the game designer (TXT and XML-like files) and the artist (all media assets), storing all files on a server with version control ensures everyone has access to the latest files, and it automatically upgrades the prototype when a new file version is checked in.

So Go Prototype!

Prototyping, as the first step in a rational game production process, should be and will be a key economic success factor in game industry.

Prototyping helps monitor and reduce many risks :

- Markets risks by testing the gameplay before hand.
- Technical risks by testing out techniques chosen for a game.
- Production risks by helping dimensioning and planning the whole production process.

Trying things out early also helps uncover unforeseen difficulties and risks before they get in the way of the final production.

For the publisher, prototyping acts as a warranty: a proof that the development team is able to handle the project, as well as a "screen test" to figure out what the player's reaction will be before blindly investing large amounts of money.

Designed and developed with these challenges in mind, Virtools Dev provides accessible graphic-user-interface-based tools backed-up by powerful rendering and behavior engines to efficiently prototype gameplay and immediately test a games' feasibility and appeal. Virtools Dev is the prototyping tool of choice to combine rapid development cycles with incredible end results.

Virtools SA
93, rue Vieille du Temple
75003 Paris - FRANCE
Ph: (+33) 1 42 71 46 86 / Fax: (+33) 1 42 71 86 53

www.virtools.com

Virtools Canada Inc.
1250, Bvd Rene Levesque Ouest/Suite 2200
Montreal QC H3B 4W8 - CANADA
Ph: (+1) 514 989 3126 / Fax : (+1) 514 989 3127