

# InvenTcl: Making Open Inventor Interpretive with Tcl/[incr Tcl]

Sidney Fels, Silvio Esser, Armin Bruderlin, Kenji Mase

ATR Media Integration & Communication, Seika-cho Soraku-gun, Kyoto 619-02, Japan  
email: fels@mic.atr.co.jp

Open Inventor is an object oriented 3D graphics toolkit written in C++. Because Open Inventor is written in C++, typical user code development consists of a program/compile/debug iteration cycle. In this sketch, we introduce *InvenTcl* which is an interpretive version of Open Inventor. Our interpretive version of Open Inventor uses the interpretive language Tcl, (Tool Command Language) [4], and its object oriented extension, [incr Tcl] [3], for the conversion. The advantages of InvenTcl include: script-able and direct manipulation of objects in an Open Inventor scene, easy prototyping of 3D graphics and animation, and low-bandwidth communication of 3D scenes and animations (using scripts).

There are three command types provided by InvenTcl which correspond to the main functions available in Open Inventor: object creation commands, object interaction commands and animation commands. For object creation there are command names for instantiating objects. These commands have the same name as the class names in Open Inventor, e.g. in InvenTcl there is a command called `SoSeparator` which creates a new separator [incr Tcl] object with corresponding methods to the public methods defined in Open Inventor for an `SoSeparator`. For interaction, InvenTcl has binding mechanisms to allow Tcl procedures to be called when objects in the 3D scene are selected. For animation commands, InvenTcl provides access to animation functions found in Open Inventor (i.e. engines and sensors). To illustrate creation commands with an example, the following code shows how a simple scene graph is created interpretively:

```
SoSeparator >root>separator1
SoMaterial >root>separator1>material1
SoCube > >root>separator1>myCube
```

This series of commands adds an `SoSeparator` node *separator1* to the root node, an *SoMaterial* node to *separator1* and an *SoCube* node to *separator1*. Notice, that we use the ‘>’ notation to specify parent/child relationships. This is consistent with how the most common toolkit used with Tcl, Tk [4], works.

There are four main technical problems to overcome to make Open Inventor interpretive:

1. accessing Open Inventor's object functions and the object's public methods and variables from the Tcl interpreter,
2. Open Inventor event management within Tcl/Tk,
3. binding Open Inventor objects to Tcl procedures and interaction modes,
4. synchronization of Open Inventor and Tcl processing.

A utility called Itcl++[2] is used to convert Open Inventor object's methods into [incr Tcl] classes. Event management is done using a handler callback mechanism available in Tcl. Implementation of event management within Tcl does cause some performance penalty. Object binding is accomplished using callback mechanisms and Open Inventor selection utilities. Object binding is an important feature of InvenTcl. By using bindings it is possible for Tcl procedures to be called as a result of interaction with Open Inventor objects. Not only does this allow for fast prototyping of direct manipulation interfaces with 3D objects but it also allows designers to easily create novel 3D interfaces for controlling other systems using the "glue-language" properties of Tcl. Synchronization is performed using a semaphor. Implementation for accessing Inventor objects' public variables has not been done in a generalized fashion yet. Further, making Open Inventor interpretive has come as an offshoot of another project[1]; hence, we still have some commands which are tailored to our application.

## References

- [1] BRUDERLIN, A., FELS, S., ESSER, S., AND MASE, K. Hierarchical agent interface for animation. In *accepted for Workshop Proc. of the International Joint Conference on Artificial Intelligence (IJCAI'97)* (August 1997).
- [2] HEIDRICH, W., AND SLUSALLEK, P. Automatic generation of Tcl bindings for C and C++ libraries. In *Proc. of the Tcl/Tk Workshop* (July 1995).
- [3] MCLENNAN, M. [incr Tcl]: Object-oriented programming in Tcl. In *Proc. 1st Tcl/Tk Workshop* (University of Berkeley, CA, USA, 1993).
- [4] OUSTERHOUT, J. K. *Tcl and the Tk Toolkit*. Addison-Wesley, New York, 1994.