The IMI SDK: component parts

Core Technology

1) An *infrastructure* for multiple input device inputs using the structures in Win32.

- *This is the primary feature in enabling an independent multiple input system. This feature and Feature 2 (the callback system) form the core of the IMI SDK infrastructure. The programmer will now see an independent and programmable stream of data from each input device.*

2) A *callback system* for developers to integrate and use multiple input devices in proprietary products.

- *The callback system provides the second half of the core infrastructure. Programmers will now be able to assign functionalities from the various input devices to appropriate aspects of the licensing application.*

Add-On Features

3) *Cursor rendering* for the multiple input devices, for appropriate 'listening applications'.

- *The first of three 'add-on' features, cursor rendering enables the assignment of individual and color-coded cursors for each of any number of input devices. The cursors can be rendered as left or right hands, left or right pointers, etc. Each cursor represents the GUI movement of an independent data stream in the operating system.*

4) An (unseen) overlay *grid* that aligns the cursors on the GUI screen, individually assignable and configurable for each input and cursor.

- *The second of the add-on features, the GRID is enabled only when the licensing application is open. Unlike most application grids, the IMI Grid retains granularity regardless of the focus of the application, allowing users to view an electrical socket with the same grid granularity as they view a house (CAD example). The GRID is individually configurable for each independent multiple input device.*

5) A *system* to use a second input device and associated cursor to navigate the GUI via *automated association with GUI objects* such as icons, buttons, and/or control points. (For example: automated association with pre-selected icons on the desktop as the 'left hand' [or non- dominant] mouse is moved.)

- *Objects represents a significant forward leap in the world of input automation. Each input device can be pre-assigned to configurable 'Object Sets'. The Object Sets can be selected from any 'listening application'. This would allow, for example, a user of an image editing suite that included a graphics application, a vector graphics application, and video editing platform to create a sub-group set of user-defined buttons that linked across platforms to associated functions. It would also allow a stock trader, for example, to link functions or data from across a wide array of data screen information into convenient sub-groups, to be accessed by any chosen input device. The programming possibilities of the Objects system are virtually endless.*