Technical Achievements

**Author(s)**
Towne, Douglas, M., Ph.D.

**Performing Organization Name(s) and Address(es)**
General Analysis, Inc.
1120 Pope St., Suite 201C
St. Helena, CA 94574

**Sponsoring/Monitoring Agency Name(s) and Address(es)**
Office of Naval Research
Ballston Tower One
800 North Quincy Street
Arlington, VA 22217-5660

**Abstract**
The tasks proposed under this agreement have been completed as documented via four quarterly progress reports. The two products, DIAG and ReAct provide all the development and delivery functions that were to be developed under this agreement.

The resulting system architecture exploits a single device model to support instruction of basic concepts, performance of procedures, and fault diagnosis, as well as aiding maintenance in the field.

**Subject Terms**
diagnostic reasoning, intelligent tutoring, model-based instruction, simulation engine
Technical Achievements

N00014-00-3-0014

Summary

The tasks proposed under this agreement have been completed as documented via four quarterly progress reports. The only changes of significance are that 1) the simulation engine formerly named InSight has been renamed ReAct, and 2) DIAG now includes the functionality that was to be in a separate module named DIle. Thus, the two products DIAG and ReAct provide all the development and delivery functions that were to be developed under this agreement.

The resulting system architecture exploits a single device model to support instruction of basic concepts, performance of procedures, and fault diagnosis, as well as aiding maintenance in the field.

![Diagram showing Concept training, Operator training, Maintenance training, and Field Aiding connected to System Model in DIAG.]

The Simulation Engine

ReAct provides a unique ToolBook-based simulation engine that supports the maintenance of either manually-authored cause-effect relation specification or entirely automated detection of cause-effect relationships, from the underlying rules of objects in a device model. This permits the developer of a device model to specify object behaviors in terms of variables named by the developer, and to leave the maintenance of the device model to ReAct, as a user interacts with it.

The rotation capabilities within ReAct were extended to provide graphical rotation of complex groups about any center of rotation.

Model Building and Navigating Tools

The DIAG product now provides a complete set of tools for creating hierarchical device models. These permit the developer to produce a new, deeper, level of a system representation with a single mouse click on any graphical element of a screen. DIAG then
creates a new screen dedicated to the more detailed representation of that element. At run
time, DIAG automatically changes the mouse cursor when the mouse enters any
graphical element that possesses a deeper representation, and it presents that next level
when the user selects it. DIAG also maintains a ‘Browse History’ list that displays all
levels of the hierarchy currently being visited, facilitating return to any higher or lower
level with a single mouse click.

Finally, a function was produced that allows the developer to embed non-hierarchical
links in any device model. Thus, one could present a button that steps the user through a
procedure or a sequence of concepts, within the context of the device model.

**Procedure Presentation System**

Functions were developed that permit DIAG to 1) present the instructions for performing
a procedure, via text and graphics, 2) demonstrate a procedure via animated graphics and
linked text, and 3) support a learner in practicing the task. Of particular importance is that
these three instructional phases rely on a single underlying ReAct model of a device or
process. Thus, the developer need produce only one ReAct simulation, then can relatively
easily produce the screens that present the instructions, demonstrate performance of the
task, and intelligently observe the learner as he or she attempts to accomplish the task on
the simulation.

**Authoring and Delivering Intelligent Diagnostic Instruction**

The DIAG system was produced entirely in ToolBook OpenScript (with support from the
ReAct simulation engine, also cast in OpenScript), thereby permitting it to function
completely and natively within the Windows/NT operating system environment. Thus,
the development and delivery of diagnostic applications is accomplished entirely within
Windows/NT, and requires no conversion operations of any kind. The DIAG developer
can switch between development mode and user mode with a single mouse click,
allowing immediate testing and evaluation of the instructional presentation.

Four additional advantages of this approach are that
1) all DIAG applications can be delivered easily from Web-based sites,
2) end users need purchase no additional software, as the ToolBook runtime system is
available at no cost,
3) any ReAct simulation can be presented in a window within a DIAG application,
   thereby eliminating the need to embed simulations of generic equipment, such as
   oscilloscopes and multimeters, within the more specific application, and
4) DIAG applications can employ any of the hundreds of Windows-based (OCX)
   programs, such as text-to-speech systems, dedicated animation systems (such as
   Macromedia Flash), or more specialized multimedia presentation systems.
Field Performance Aiding

The DIAG user interface was extended to support aiding of field maintenance functions, with no additional authoring requirements. This interface accepts observed symptom information from the end user and employs the extant fault reasoning process to progressively reduce the suspicion set.

System Support

An interactive application was developed, in ReAct, to document and store examples of the various ReAct functions. This software module contains the full specification of the ReAct animation/simulation functions, as well as operable examples that can be copied and pasted to new applications.

In addition, a User’s Manual for applying DIAG was produced, along with a power supply application.