SmartArrays APL Translator
Frequently Asked Questions

The SmartArrays APL Translator, available in April 2004, provides a facility for automating the conversion of existing APL programs into widely used compiled languages. It works in conjunction with the SmartArrays Software Development Kit, which provides an APL-like array facility for compiled languages.

Where is the Translator useful?

The Translator is designed with these goals in mind:
- Preserve the intellectual capital in legacy APL applications by allowing their core computations to be migrated to a compiled language.
- Permit the use of these computations in new places – for example as production systems in organizations where the IT infrastructure does not support APL.
- Allow developers to write data-intensive and numerical algorithms in APL, and then deploy those algorithms in a compiled language.

Which versions of APL does the Translator support?

The translator is available for IBM APL2 although it can work with other APL systems, notably APL+Win (APL2000) and Dyalog APL (Dyalog Ltd) where these implementations match the IBM implementation.

What languages can APL be translated into?

The Translator transforms APL into C++. Future versions will support other languages like Java and C#.

How does the APL Translator work?

The Translator is a set of APL programs packaged in an APL transfer file (.atf), such as one produced by the )OUT command in APL2. You use it in the APL environment by copying the Translator into your workspace with )PIN or )IN and running the translate functions. You can translate individual APL functions or an entire workspace.

The output of the translator is C++ source files containing calls to SmartArrays for C++. These can then be incorporated into a program that uses compiled SmartArrays instead of the APL interpreter to perform the array operations.

Can entire APL applications be translated?

The Translator applies only to the computational core of an APL application – the array operations that structure data and calculate on that data. With a few exceptions, all the APL primitives are converted. (A notable exception is the "execute" function.) System functions and system variables (names that start with the Quad symbol) are not translated except for QuadIO and Quad output.

The assumption is that translated APL functions will be incorporated into new applications that are developed in the target environment. For example, you might want to develop a web service that provides
on-demand calculations based on a collection of APL functions. You would use a C++ development environment and web service components to set up a skeleton service that exposes a SOAP-callable interface and converts data arguments into SmartArrays arrays. Then you would plug in the C++ functions generated by the translator to perform the computation.

**Will programs run faster after translation?**

There probably will not be a great deal of difference. The SmartArrays array engine uses array algorithms similar to those used inside APL interpreters, so the translated programs will be performing operations that are much like those of APL.

**Is it easy to fine tune the translated program to get faster execution?**

Yes. Scalar integer computations in the program can be translated to C++ scalar integer operations by adding a translator directive to the APL program. This will yield very large speedups to scalar-oriented APL code.

A C++ programmer can perform some optimization on the translated program with the help of the APL programmer. For example, loops in the APL program could be replaced with C++ control structures like “for” and “while”. Common APL idioms could be replaced with their SmartArrays equivalent for increased performance. For example, “shape().shape()” could be replaced with “rank()”. There are also a number of high-performance operations in SmartArrays that are not found in APL, and exploiting these can further increase speed.

**What are the runtime requirements for translated programs?**

Translated programs use SmartArrays, therefore the SmartArrays engine must be installed on any computer where they execute.

**What platforms will translated APL programs run on?**

Because the resulting programs require the SmartArrays library to execute, they will run on platforms for which SmartArrays is available. Contact SmartArrays for a current list of supported platforms.

**Do I need to license SmartArrays in addition to the translator?**

Yes. The SmartArrays SDK and any deployment licenses are licensed separately from the translator. At least one developer in your organization will need to have the SDK in order to compile and test the program, and you will need the appropriate SmartArrays deployment license to put the application to use. The sales department at SmartArrays can advise on what you will need based on how you plan to deploy the converted APL programs.

**What does the translated code look like?**

It’s quite readable and looks surprisingly similar to the APL source. Here is one simple sample:

```apl
ζSmA_TranslateLine 'A*B*C'
  A = (B*C.transpose());
```

A, B, and C are SmartArrays. Common arithmetic and comparison operations in C++ (like ‘*’) are overloaded so they apply to SmartArrays.

**Is there a trial version of the translator?**

Yes. *Trial versions of the translator and SmartArrays for C++ are available on request.* Send an email request to sales@smartarrays.com.