

redFOX Fast Facts

Version 1.9
16 January 2006

What is redFOX?

redFOX is a high performance distributed object infrastructure for embedded systems being developed by REDPLAIN.COM.

- It is smaller and faster than any comparable product.
- It is very easy to use because it makes C++ objects remotely accessible without need for IDL.
- It preserves thread priority in remote invocations.
- It supports heterogeneous processors and operating systems.
- It runs on Linux and Windows and is portable to any operating system with threads, semaphores and a simple message transport.
- It works with g++ 3.0+ and can support alternative C++ compilers.

How is redFOX used?

Just write a distributed application in C++ and reference remote objects using redFOX's *drefs*, which are conceptually close to C++ references.

The redFOX code generator works with your own compiler to inject middleware where appropriate. The compiled code is linked to the redFOX runtime library.

What applications are redFOX suitable for?

redFOX suits a broad range of distributed embedded applications and parallel processing applications.

It excels where other middleware solutions suffer from excessive software overhead or excessive bandwidth consumption.

Embedded applications include Aerospace, Automotive, Consumer Electronics, Game Console, Storage and Telecom industries.

redFOX supports load balancing of complex applications by allowing objects to be easily assigned to different processors.

How fast is redFOX?

Local method calls have an overhead similar to a virtual function call. Remote calls are done in a time that matches the best hand crafted message passing. Messages are smaller than 20 bytes in many cases, which minimizes software overhead and latency on both fast and slow communication channels.

How does redFOX contribute to distributed application portability, hardware independence and software re-use?

redFOX supports the implementation of distributed object models, which inherently de-emphasizes hardware architecture. It minimizes the overhead of local invocations as well as remote invocations. This supports the use of fine grain distributed object models – designs composed from small objects with simple functionality that are independent of hardware architecture.

Such object models can be efficiently deployed on a variety of hardware architectures. It is the deployment, rather than the object model itself, that is influenced by hardware architecture.

Simple classes from fine grain object models are more re-usable than the complex classes that become necessary when the overhead of method invocation is not minimized.

How does redFOX reduce the cost and time of software development?

redFOX automatically generates middleware that would otherwise have to be hand coded

to achieve the same performance. Such hand coded middleware routinely accounts for a large proportion of a software development project, running to thousands of lines of middleware per class. One company developing embedded systems estimates that redFOX will eliminate 25% of its source code.

In addition to reducing implementation effort, redFOX improves productivity by allowing developers to work at a higher level of abstraction and by making design changes easy to implement.

How does redFOX improve software quality and reliability?

redFOX improves quality and reliability directly and indirectly.

It does it directly by replacing thousands to millions of hand coded middleware with less error prone automatically generated code.

It does it indirectly by facilitating software re-use, which allows field proven application code to be deployed in new products.

How does redFOX help with building High Availability and Fault Tolerant systems?

redFOX is an asset in High Availability systems for two reasons. First, low overhead invocations allow simpler application architectures with less concurrency than high overhead alternatives. Second, redFOX minimizes the communication cost and software overhead of maintaining standby resources. This helps avoid High Availability impacting the application architecture.

What about security?

redFOX uses pluggable transports. An application can use library encryption and authentication or supply its own on a connection by connection basis.

What hardware and software is redFOX compatible with?

redFOX can run on a variety of processors, operating systems and message transports. It works across heterogeneous processors and

operating systems. Initially, gnu gcc 2.95 and successors are supported and all its supported processors. Windows, Linux and OSE will be supported initially.

Can redFOX be used with CORBA?

Although redFOX does not adhere to the CORBA standard (allowing it to be lighter, simpler and *much* faster), it is possible for redFOX and CORBA to co-exist in one application. redFOX objects can invoke CORBA objects and vice versa.

Tell me more.

See www.redplain.com for a product description. Contact info@redplain.com with questions.