

RINOVA - Re-Engineering NCS Subsystem

Customer is a world leader in manufacturing office automation equipments such as special purpose printers, Multi-Functional printers etc.

Business Objective

Client's Multi-Function Printer (MFP) is an integration of printer, fax, scanner and copier. Network Control Services (NCS) is a major subsystem of the printer control software of client's multifunction printers as well as other printer models. It is responsible for handling various network protocols supported by the printer. NCS with a code base of about 250,000 LOCs acts as a bridge between the driver/physical layer of different types of network interfaces and applications like print, fax etc.

The customer normally develops new protocol daemons every year for their printer to support new protocols, which are getting standardized in the industry. Due to the complexity of the NCS subsystem, it was taking too much time for the client to develop a new protocol daemon. This was a serious concern for the customer and because of this, they were looking for a solution to reduce the development time of new protocol daemon by improving the modularity and maintainability of the NCS subsystem.

System Overview – Challenges & Solution

The basic system architecture is as below:

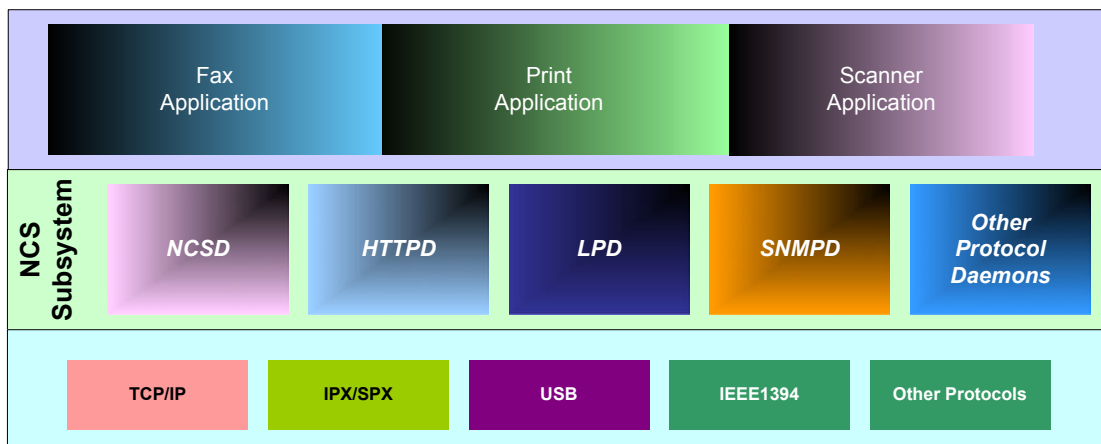


Figure 1: High Level Architecture of NCS subsystem

As depicted in figure 1, NCS is the middle layer of the printer software. NCS subsystem is composed of a number of daemons, each providing a different functionality. The different components of NCS are classified into the following categories:

1. Printing Protocol Daemons
2. Management Daemons
3. Communication Daemons
4. Other Components

The Challenges

Re-engineering NCS subsystem posted the following challenges in front of us.

1. Understanding the huge source code base of 2500 KLOC without any documentation
2. Interact with different engineers in the development team and understanding the real difficulty faced in the development of new protocol daemon and understand their ideas on required system architecture.
3. To propose an architecture, which enable the client to reduce the development time of the new protocol daemon and provides better maintainability.

Solution

Based on a detailed study and evaluation of NCS sub-system, the main problems resulting in complexity of NCS were found to be the following:

- Message sequencing achieved in a complex way
- Protocol daemon developer needs to understand the complexity of the message sequence to develop a new protocol daemon.
- Different protocol daemons have different way of implementation of same message sequences
- Complex way of data exchange is used
- Protocol Daemon uses a lot of queues for message reception
- Threads are used where not required
- Multiple communication mechanism between processes
- Difficulty in conveying the design to protocol development engineers

New Design

The main purpose of the redesign was to reduce the overhead on protocol daemons and to make the messaging scenarios more standardised. This necessitated the development of a interface library to handle the messaging scenarios such that the protocol daemon need not worry about message handling.

The new design included:

1. Development of a NCS Interface Class Library (NICL)
2. Development of a File Interface Library

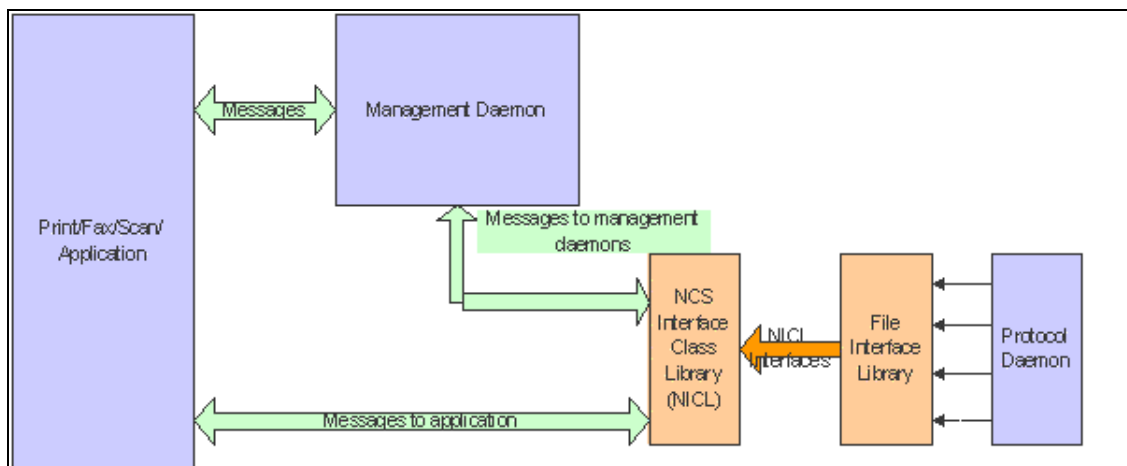


Figure 2: High-level architecture of the newly designed system

NCS Interface Library:

NCS interface class library, is a set of classes. Protocol daemon's interface to NCSD and application is made through NCS interface class library (NICL). NICL takes care of all the message scenarios between NCSD and application. With this design, a protocol daemon can be developed by calling the interfaces of the NICL without knowing the messaging scenarios. Thus overhead on the protocol daemon is reduced considerably. Interface between NCSD and application is retained as such.

File Interface Library:

Protocol daemons are usually developed from open source code, which uses the general system calls. The prime aim behind the design of File Interface library is to maintain the open source code changes to a minimum. File interface is a generic layer, which uses the NICL. The file interface design provided an easy mechanism to use the different application (print, fax, scanner etc) of the NCS subsystem. It provides interfaces to access the NCS subsystem in the simplest possible way.

Advantages to the Customer

Our solution enabled the client to save significant amount of time, effort and cost in developing new protocol daemon. The customer is using our solution for the development of new protocol daemon. Since our solution co-exist with the existing legacy system, customer could retain their existing system and develop new protocol daemon based on our solution.

Development Environment

Target OS: NetBSD
Host OS: FreeBSD
Languages: C and C++
Debugger: Gdb

Design Methodology Using Object Oriented Design and state-machine for implementing messaging sequence