# Aspects of 802.16 Network Management

by James Wiebe
for Dr. Tepe
88-590-17 Wireless Communication Systems, W2005
April 29, 2005
University of Windsor

**Abstract –** Network management techniques used in the IEEE 802.16-2001 [5] standard and the algorithms built to use them, are described. A C++ dialog-based program illustrates the Dynamic Service Flow state Machine.

## Summary

802.16 Network entry and initialization (when a SS is powered-up) is composed of the following phases:

a) Scan for a downlink channel
b) Establish synchronization with the BS
c) Obtain transmit parameters (from UCD message)
d) Perform ranging
e) Negotiate basic capabilities
f) Authorize SS and perform key exchange
g) Perform registration
h) Establish IP connectivity
i) Establish time of day
j) Transfer operational parameters
k) Set up connections

At the end of this, the MAC transport connections for the service flows between the BS and SS have been established.

Service flows may be created, changed, or deleted. This is accomplished through a series of MAC management messages referred to as Dynamic Service Addition (DSA), Dynamic Service Change (DSC), and Dynamic Service Deletion (DSD). DSA Messages create a new service flow, DSC Messages change an existing service flow, and DSD Messages delete an existing service flow. Figure 3 shows the Dynamic Service State Machine flow, and Figure 4 shows the C++-programmed state machine illustration.

## 1. Introduction

The 802.16 standard is a large one, taking up 349 pages in total. It specifies the operation of the OSI Physical layer and a lower part of the Data Link layer comprising a wireless metropolitan area network (MAN) standard. It is intended to scale to larger areas than does the popular 802.11 wireless Local Area Network (LAN) standard, which has been implemented as “WiFi”. Its Service Specific Convergence Sublayer is designed to support ATM (Asynchronous Transfer Mode) cells as well as packets; therefore one of its applications could be to support IP (internet protocol). This has the potential of realizing the dream of ubiquitous portable high-speed WWW (World-Wide Web) access. An industry group is currently working on implementing 802.16 as “WiMax” [3].

Although the 802.16-2001 standard has been updated and “obsoleted” by the 802.16-2004 standard, the latter is not yet available for public access via the IEEE “Get 802” program [1]. These standards are made publicly available via this program six months after release and the new version was completed only in October 2004, making it not quite timely for this paper, which is due in April 2005.

The following diagram gives an overview of the scope of the 802.16 standard, taken from Figure 1 in the 802.16-2001 standard ([5], pg 2):

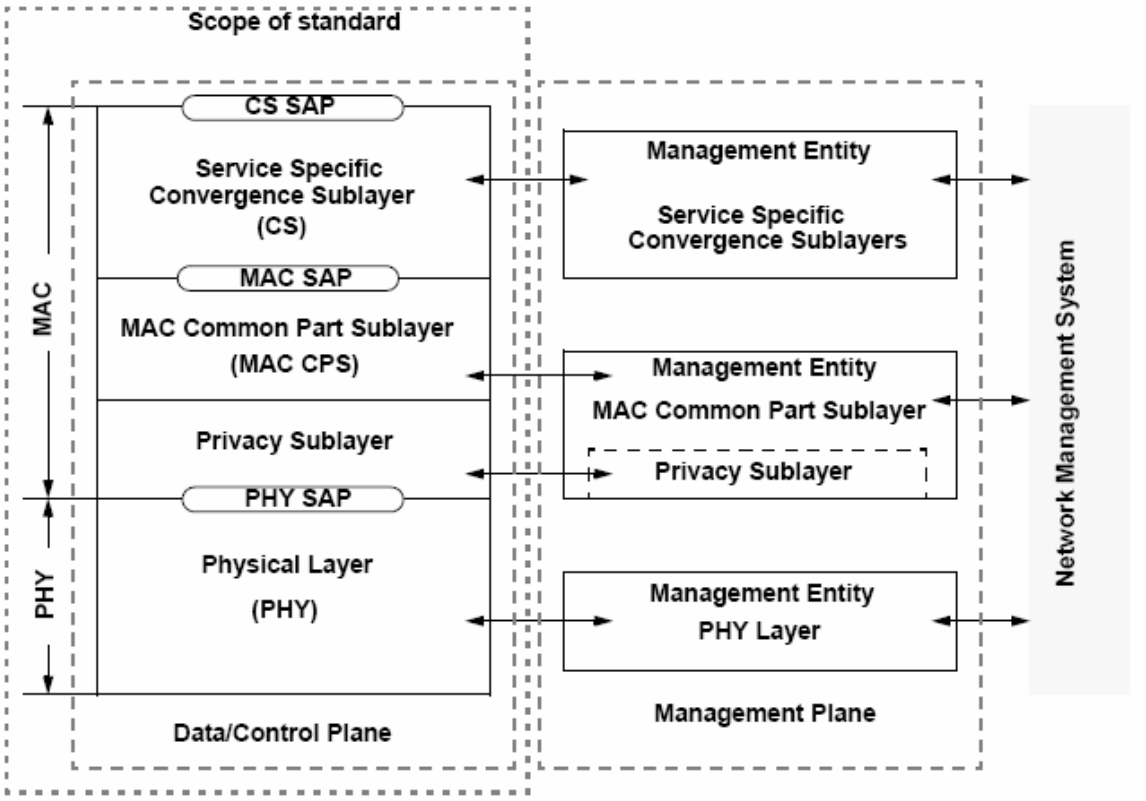


Figure 1. IEEE std. 802.16-2001 protocol layering.

"SAP" stands for "Service Access Point", each of which forms a conceptual "doorway" between an adjacent pair of sublayers. Each layer communicates with the same layer in the other entity and information passed down to it from the next higher layer is treated as "black-box" data that is only "looked at" at most in order to assure that it cannot interfere with the operation of the current layer. It can be amalgamated or split up into different packets before being passed down to the next lower layer. Data is received from above in the form of SDUs (Service Data Units) and processed by the current layer into PDUs (the Protocol Data Units of the current layer), which become in turn the SDUs of the next lower layer.

The overall architecture is that of one Base Station (BS) acting as a server for one or more Subscriber Stations (SS).

This paper focuses on the operation of two of the MAC-level (Media Access Control layer – the lower portion of the link layer) Network Management functions of 802.16: Network entry and initialization, and Quality of Service.

## 2. Network Entry and Initialization

It is the responsibility of the SS to establish communications. Each SS comes from the manufacturer with a built-in 48-bit MAC address and an X.509 certificate used to authenticate the SS (discussed in the Privacy section of the standard [5]).
The Network Entry and Initialization procedure can be divided into the following phases:

l) Scan for a downlink channel
m) Establish synchronization with the BS
n) Obtain transmit parameters (from UCD message)
o) Perform ranging
p) Negotiate basic capabilities
q) Authorize SS and perform key exchange
r) Perform registration
s) Establish IP connectivity
t) Establish time of day
u) Transfer operational parameters
v) Set up connections

**2.1 Scanning and synchronization to the downlink**

The SS keeps its latest operational parameters in nonvolatile memory and tries those first. If those don't work or it has none, it scans its entire valid bandwidth continuously until it finds a valid signal. The MAC layer gets a PHY (physical layer) indication when this occurs and then the MAC attempts to acquire the channel control parameters for the downlink and then the uplink. The MAC achieves synchronization when it has acquired at least one DL-MAP message. It deems its synchronization lost when time delays pass without the reception of a DL-MAP or DCD (Downlink Channel Descriptor) message, at which time it tries to re-achieve synchronization.

**2.2 Obtain transmit parameters**

The SS waits to receive a UCD (Uplink Channel Descriptor) message on the downlink. If none is received within a timeout, it reverts to scanning for a new downlink channel. When it does get one or more UCDs, it checks their parameters to determine whether it may use one of the uplink channels. If it may not use any, it reverts to waiting for UCDs, from whence it may revert to scanning for downlink channels.

The BS periodically resends UL-MAP and DL-MAP messages during normal operation in order to keep the SS appraised of any changes to the locations of data bits, maintenance bits and request bits, etc. in each MAC frame.

**2.3 Perform ranging**

Ranging is the process of acquiring the correct timing offset such that the SS's transmissions are aligned according to the BS's processing. The SS will perform initial ranging at least once on a channel ID (one channel ID per UCD). If not successful, then the next channel is tried. If unsuccessful with all channel IDs, the SS reverts to scanning for a new downlink channel. In ranging, the SS sends a RNG-REQ message, beginning at

minimum power level, and subsequently at incrementally higher power levels until successful (if at the highest power level, it steps back to the minimum power level). If contention with another SS occurred, the SS will wait a random "back-off" time before resending its RNG-REQ message. Otherwise at this point the ranging attempts continue indefinitely. When successful, the BS returns a RNG-RSP message containing the CIDs (connection IDs) that the SS will use, an RF power level adjustment, a frequency adjustment and a timing adjustment. The SS will then send a second RNG-REQ message using the CID provided and the BS will reply with a second RNG-RSP message containing further fine-tuning information. These exchanges will be repeated until the BS sends a Ranging Successful notification or the BS aborts ranging, in which case the SS determines that ranging was not successful.

### 2.4 Negotiate basic capabilities

After completion of ranging, the SS sends an SBC-REQ message listing its capabilities (each set to "on"). Basic capabilities include only certain physical parameters and bandwidth allocation. The BS returns an SBC-RSP message indicating the capabilities that it shares with the SS via the mechanism of turning "off" the indication of any capabilities that were sent to it as "on", that it does not share with the SS.

### 2.5 Authorize SS and perform key exchange

The SS is authenticated by the BS and symmetric keys are shared using public-key cryptography. Description of this can be found elsewhere [6] and is beyond the scope of this report.

### 2.6 Perform registration

The SS must register itself with the BS in order to become manageable; it receives a Secondary Management CID for this purpose in response to a REG-REQ message, i.e. in a REG-RSP message. If retries of this become exhausted the SS returns to the point of physical synchronization and restarts from the point of trying to acquire MAC synchronization (i.e., getting the downlink map message). In response to the REG-RSP message, the BS expects a TFTP-CPLT (Trivial File Transfer Protocol Complete) message. If it does not receive this after a timeout, it will deassign the management CIDs from that SS (and the SS will have to return to the point of physical synchronization).

### 2.7 Establish IP connectivity

The SS may include the IP version number it supports in its REG-REQ message. If it includes no indication, the BS will assume that it supports IPv4 only, otherwise the BS will return a command to the SS to use the indicated version for its secondary management connection. Omission of this command is interpreted by the SS as a command to use IPv4 for the secondary management connection. The SS will then use DHCP (IETF RFC 2131) in order to obtain an IP address and parameters needed to establish IP connectivity.

### 2.8 Establish time of day

This is required for the time-stamping of logged events and is done to the nearest second, over the secondary management connection. The request and response is done via UDP and the response is in the form of Universal Coordinated time. This is combined with the time offset established via DHCP to calculated the local time.

### 2.9 Transfer operational parameters

The SS then downloads the SS Configuration File using TFTP. The configuration file always contains the following:
a) SS MIC (Message Integrity Check) Configuration Setting
b) End Configuration Setting (end of data marker).

The configuration file may also include:
a) Software Upgrade Filename Configuration Setting
b) SNMP Write Access Control (multiple)
c) SNMP MIB Object (multiple)
d) Software Server IP Address
e) Pad Configuration Setting (as needed to hit 32-bit boundary in file)
f) Vendor-Specific Configuration Settings

If present, these must be supported by all of the BS's SSs.

### 2.10 Set up connections

The BS sends DSA-REQ (Dynamic Service Flow Add Request) messages to the SS in order to set up service flows. The SS responds with DSA-RSP messages. This establishes MAC layer transport services for packets, one-way only per Service Flow.

## 3. Quality of Service

A service flow, as identified by its CID, has QoS (Quality of Service) parameters associated with it and which characterize it. QoS primarily depends upon the air interface, and QoS parameters include latency, jitter and throughput assurances. The service flow is the primary mechanism by which QoS parameters are associated with a stream of data. A service flow is a unidirectional MAC-layer transport service; thus two at least are required; one for uplink packets to the BS and another for downlink packets to the SS.

A service flow is characterized by the following attributes: the Service Flow ID, the Connection ID, the "Provisioned QoS Parameter Set" (provisioned by general network management techniques), the "Admitted QoS Parameter Set" (by reservation of resources on the part of the BS and SS), the "Active QoS Parameter Set", and the Authorization Module (within the BS that decides whether to allow any change in QoS parameters).

Service flows can be preprovisioned or new flows can be activated dynamically. Preprovisioned service flows can be static so that change requests (DSC-REQ messages) are refused (they can only be activated or deactivated).

### 3.1 Dynamic Service Flow Creation

Service flows may be created via the DSA-REQ message by the BS or the SS. A three-way handshake is used, and both an uplink and a downlink service flow is created. Here is a diagram of the protocol for the BS to initiate the request (Figure 2).

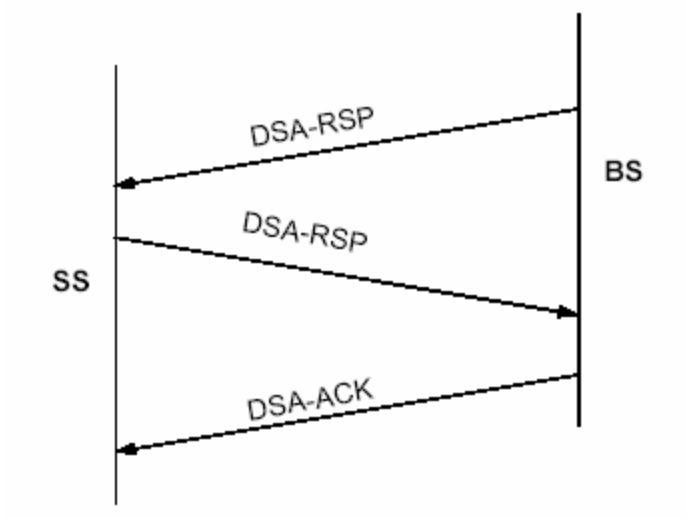


Figure 2. DSA message flow – BS-initiated

### 3.2 Dynamic Service Flow Modification and Deletion

Protocols are defined for modifying and deleting service flows. Service flows may be created, changed, or deleted. This is accomplished through a series of MAC management messages referred to as Dynamic Service Addition (DSA), Dynamic Service Change (DSC), and Dynamic Service Deletion (DSD). DSA Messages create a new service flow, DSC Messages change an existing service flow, and DSD Messages delete an existing service flow.

The Null state implies that no service flow exists that matches the SFID (Service Flow ID) and/or Transaction ID in a Message. In steady state operation, a service flow resides in a Nominal state. When Dynamic Service messaging is occurring, the service flow may transition through other states, but remains operational. Since multiple service flows may exist, there may be multiple state machines active, one for every service flow. Dynamic Service Messages only affect those state machines that match the SFID and/or Transaction ID. If privacy is enabled [6], both the SS and BS verify the HMAC digest on all dynamic service Messages before processing them, and discard any Messages that fail.

The Dynamic Service Flow State Transition Diagram (Figure 3) shows the top-level service flow state, driven by the DSA, DSC and DSD messages. There are six different types of transactions: locally initiated or remotely initiated for each of the DSA, DSC, and DSD Messages. This state diagram applies equally to the BS and SS, except in the Dynamic Service Flow Changing-Local state, in which only the BS can change its own local state (see Fig 3).

**State Machine**

The Dynamic Service flow state machine consists of 9 states and 8 types of events, including receipt of messages, that can trigger state transitions (see Figure 3). The creation of DSx Transactions by the Dynamic Service Flow State Transition Diagram is indicated by the notation:
"DSx-[ Local | Remote ] ( initial_input )",
where initial_input may be SF Add, DSA-REQ, SF Change, DSC-REQ, SF Delete, or DSD-REQ, depending on the transaction type and initiator.

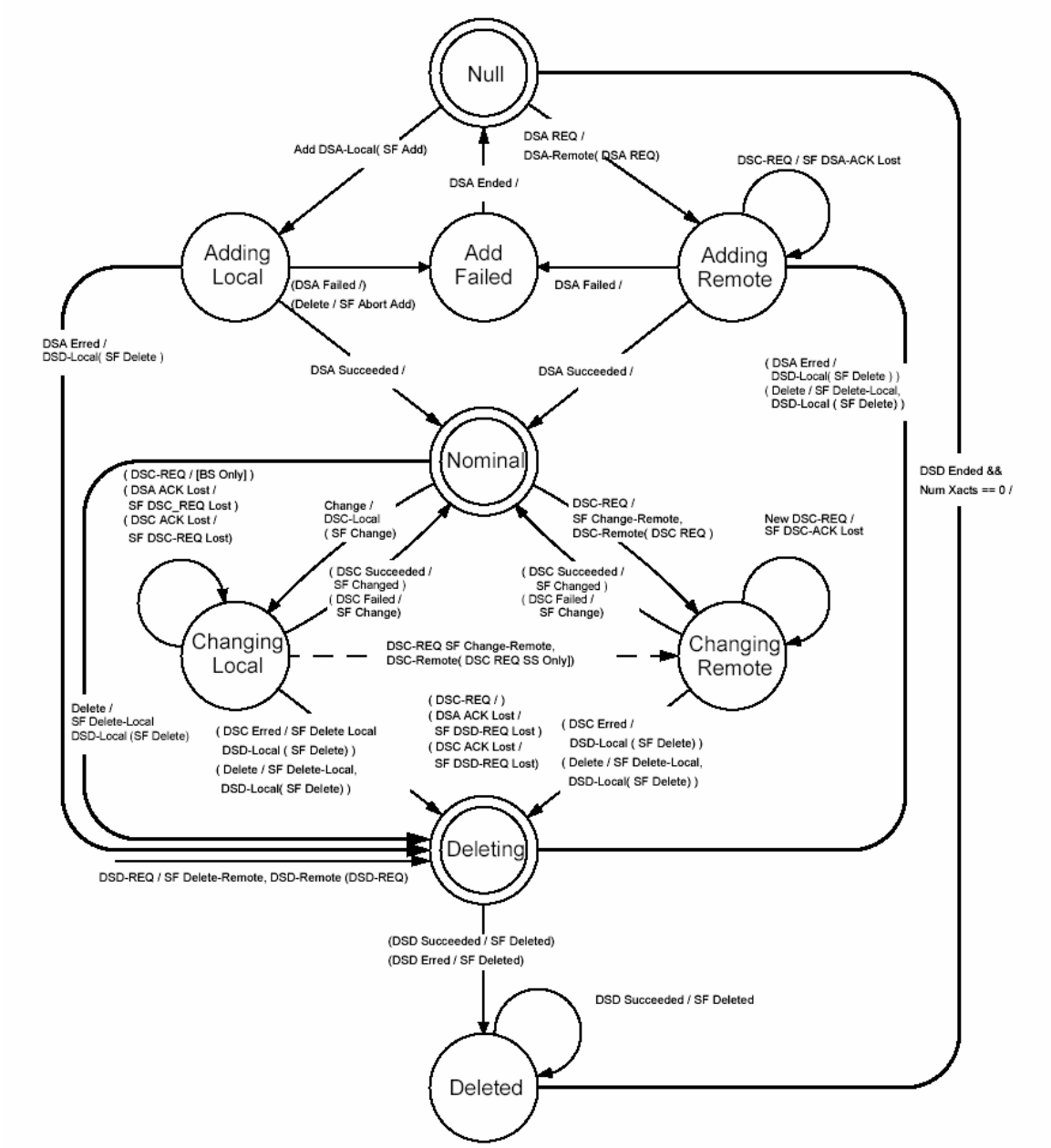


Figure 3. Dynamic Service State Machine flow

## 2.5 State Machine Illustration

Microsoft Visual C++ version 6.0 provides a convenient way to demonstrate the state machines, using its “dialog box” feature, which can be built via some clicks of the mouse [7]. The outlines of C++ classes [8] can be added to the code by the clicks of the mouse. I have implemented buttons on the dialog box, labeled with the events or messages that occur, to step the simulation through its paces, at the rate of one click per event. I have maintained the state of the state machine in a variable contained in the “dialog box” C++

class (ie in volatile memory) and its value is shown on-screen, in a “list box”, so that any state can be selected at will, which facilitates easy exploration of the state machine. A log file showing the states and messages is generated, with buttons added for its display. The log file is closed and left on the computer HD when the program is exited; it is only erased if the user explicitly clicks on “Restart and Clear Log”.

A screen capture is shown (Figure , below) of the Dynamic Service Flow state machine C++ illustration.

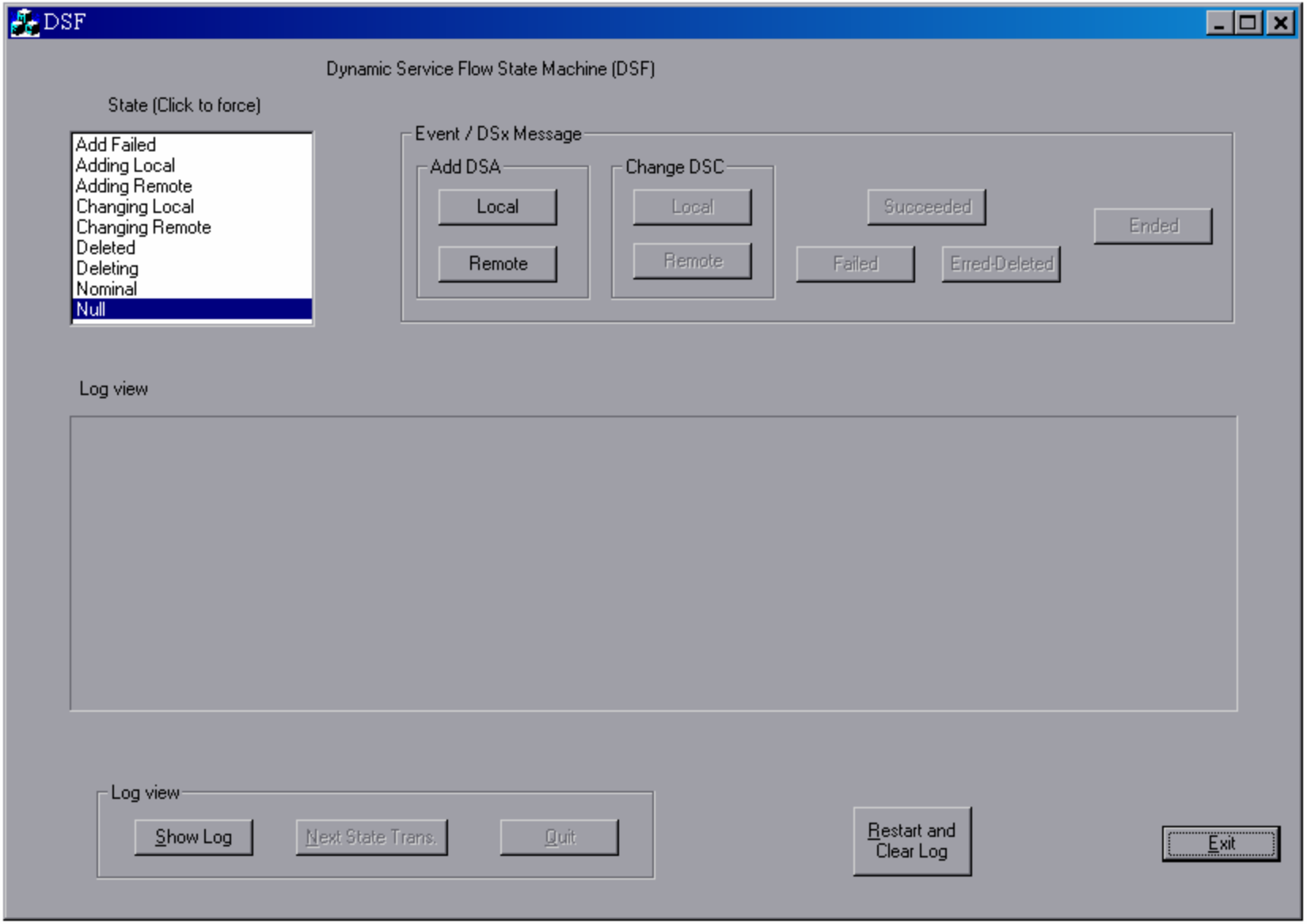
DSF
Dynamic Service Flow State Machine (DSF)
State (Click to force)
Add Failed
Adding Local
Adding Remote
Changing Local
Changing Remote
Deleted
Deleting
Nominal
Null
Event / DSx Message
Add DSA
Local
Remote
Change DSC
Local
Remote
Succeeded
Failed
Erred-Deleted
Ended
Log view
Log view
Show Log
Next State Trans.
Quit
Restart and Clear Log
Exit

Figure 4. Dynamic Service Flow state machine illustration (rotated 90 degrees to fit on page)

## Conclusions

The 802.16 standard specifies a very flexible service protocol, that has great flexibility and can become exceedingly complex in practice.

# References

**Websites** (sorted in order of base domain name)

**[1]**http://standards.ieee.org/getieee802/ - The website of the IEEE "Get 802" program. IEEE 802 standards are made available to the public, free of charge, six months after release.

**[2]** http://www.wikipedia.org – As an online free encyclopedia to which all are invited to contribute, it is rich in information on contemporary computer systems and computer-related mathematics, as well as many other areas of knowledge. It is a good place to look for explanations of acronyms the knowledge of which is taken for granted in electronic trade publications.

**[3]** http://www.wimaxforum.org – The website of the WiMax forum. "The WiMAX Forum is an industry-led, non-profit corporation formed to promote and certify compatibility and interoperability of broadband wireless products. Our member companies support the industry-wide acceptance of the IEEE 802.16 and ETSI HiperMAN wireless MAN standards."

**[4]** http://wirelessman.org – This is the website of the IEEE 802.16 working group on broadband wireless access standards.

**Documents**

**[5]** 802.16™ (2001) - IEEE Standard for Local and metropolitan area networks, Part 16: Air Interface for Fixed Broadband Wireless Access Systems, IEEE, http://www.ieee.org, New York, NY, 2002 – 349 pages.
This is the standard in all its glory, that implementers can flesh out.

**[6]** Wiebe, James, 802.16 Security and Encryption, for Dr. Erfani, 88-590-18 Network Security, W2005 April 14, 2005, University of Windsor.
My paper investigating the privacy and security aspects of 802.16.

**Books**

**[7]** Chapman, Davis, and Bates, John, Teach Yourself Visual C++ 6 in 21 Days, SAMS, a division of Macmillan Computer Publishing, http://www.mcp.com, Indianapolis, 1998, 0-672-31240-9
This book gives the follower the idea of how to get started programming with Windows in C++, i.e., the visual aspect, very quickly.

**[8]** Malik, D.S., C++ Programming - Program Design Including Data Structures, first ed., Course Technology, http://www.course.com, 2002, ISBN (2nd ed., March 23, 2004) 0-619-16044-6
This is an excellent book in explaining the detail of generic (non-visual) C++ programming to the beginner, in extremely minute detail. It is also a good intermediate-level reference.