Description

BACKGROUND OF THE PRESENT INVENTION

Field of the Invention

[0001] The present invention relates to the implementation of telecommunications services in accordance with industry standards and, more particularly to a system, method and apparatus for testing a software system using a known communication standard.

Background and Objects of the Present Invention

[0002] International standardization organizations, such as the International Telephone and Telegraph Consultative Committee (CCITT), are made up of telecommunications system operators (largely governmental agencies) which jointly agree how various communications services are to be provided so that communications are seamless across international borders regardless of which equipment vendor supplies the interconnected systems. One element of communications standards defined and specified by CCITT, are data transmission protocols,i.e.,the format of both traffic in signaling messages to be used for particular services. For example, the form of signaling messages is often described using a tabular notation with their format and binary representation being specified by a table wherein the entries are the information elements from which they are built. This technique is rather convenient when the signaling message structure of a protocol is simple and when there is no need to consider different encoding schemes to represent the same information. CCITT Recommendations covering Signaling System No. 7 and Digital Signaling System No. One (DSS1) describe most of the application protocol data units in this manner.

[0003] However, as the signaling information to be exchanged between telecommunications systems becomes more complex, the limitations of tabular notation become exceedingly clear e.g., difficulties in representing structured elements and duplication of definitions due to the mixture between the syntax of an information element and the way in which it is encoded. In order to avoid some of these problems associated with complex signaling messages, CCITT began using a different technique of describing standardized signaling messages referred to as Abstract Syntax Notation One (ASN.1). ASN.1 provides a means to describe data types as well as the value of type in an abstract manner without determining the way instances of such data types are to be represented during transmission.

[0004] The purpose of using ASN.1 to specify CCITT protocols is to have a standardized language to express types with and to provide a standardized set of rules to transform an instance of one type into a stream of bytes. A system designer can use ASN.1 to specify and design a specific data structure. An instance of the data structure can then be transformed or encoded into a stream of bytes according to a predefined set of encoding rules, such as what are known in the art as "Basic Encoding Rules" (BER). Such a stream of bytes, referred to as an ASN.1 message, is then sent on a communications channel to another application program which decodes the stream of bytes according to details specific to the application and the platform on which the application resides. Thus, two different applications written in two different programming languages and running on two different types of computers using different internal representations of data can exchange instances of structured data types instead of exchanging bytes. This system of abstract representation of types eliminates a great deal of effort since no code need be written to process the transfer format of the data.

[0005] While ASN.1 specifications for a data type are relatively simple to use, developing the programs which use the data types represented by ASN.1 specifications is a complex process. Routines must be written for encoding and decoding ASN.1 messages according to the ASN.1 specifications. The routines can be written by a compiler program which writes source code that is implanted into and becomes a part of the application program. Such compilers, however, tend to be very costly, and, consequently, the routines are often manually written. It should also be understood that because of their sheer complexity,manually written routines tend to be error-prone. Furthermore, even routines written produced by compilers have been known to include errors. Accordingly,testing a system utilizing ASN.1 specifications is extremely important to work out the various bugs.

[0006] One of the difficulties involved in testing a system utilizing ASN.1 specifications, however, is the generation of useful test data. Proper testing of an application program requires observing the input and output behavior of the application program. Because the application program will receive ASN.1 messages from another application program in the completed system, the received test data must correspond to the ASN.1 messages produced by that another application program. Furthermore,the output of the application program will also communicate with other application programs in the form of ASN.1 messages. While the other application program could be used to generate outputs for testing purposes, a benefit to using ASN.1 is that communicating application programs in a system can be individually developed. Therefore, the other application program might be unavailable or even non-existent at the time of testing the first application program. Accordingly, the test data must frequently be generated by the programmer. Furthermore, the output