ENDL
TEXAS

Date: 7 May 2003
To: T10 Technical Committee
From: Ralph O. Weber
Subject: Calls and Arguments in SAM-3

A clear and consistent nomenclature is proposed for the procedure calls contained in the SAM-3 architectural model.

Today, the terms 'call', 'procedure call', and 'remote procedure call' are used interchangeably. Likewise, the terms 'argument' and 'parameter' are used interchangeably.

The following two terms are proposed to be the only terms used in SAM-3:
- 'procedure call' — replacement for 'call' and 'remote procedure call'
- 'argument' — replacement for 'parameter'

It must be noted that all command set and SCSI transport protocol standards make reference to the procedure call and argument concepts in SAM. Most include a Notation For Procedure Calls subclause such as the SAM-3 subclause modified by Change 3. If these changes are approved for SAM-3, equivalent changes will be necessary in every SCSI standard that references SAM-3.

Revision History

r0 Initial proposal
r1 Modified as requested by the March CAP working group
- Use 'argument' not 'procedure parameter'
- Remove substitution of green and brown text for bold

r2 Modified to clarify 4.2 (The SCSI distributed service model) as requested by the May CAP working group

Specific Changes

Specific changes are proposed for SAM-3 r06.

Changes between the previous revision of this proposal and this one are marked with change bars, except for changes relating to differences between SAM-3 r05 and SAM-3 r06 which are not marked.

Change 1 [glossary]: Add a glossary entry for 'argument' and convert the current 'call' glossary entry to 'procedure call' as follows.

Add a 'procedure parameter' glossary entry as follows:

3.1.x argument: A datum provided as input to or output from a procedure call (see 3.1.x) ~~architectural abstraction~~.

Modify the 'call' glossary entry as follows (relocating it for proper alphabetical sorting):

~~**3.1.9**~~ **3.1.y procedure call:** The model used by this standard for the interfaces involving both the SAL (see 3.1.84) and STPL (3.1.99) layers, ~~An architectural abstraction~~ having the appearance of a programming language function call ~~that is used to model service interfaces~~. ~~The act of invoking a procedure.~~

18484 Preston Road, Suite 102 PMB 178, Dallas, TX 75252 214-912-1373 972-596-2775 Fx ENDL_TX@computer.org

Change 2 [Editorial Conventions]: Modify the second and third paragraphs of 3.4 (Editorial Conventions) as follows:

Upper case is used when referring to the name of a numeric value defined in this specification or a formal attribute possessed by an entity. When necessary for clarity, names of objects, ~~procedures~~ procedure calls, ~~parameters~~ arguments, or discrete states are capitalized or set in bold type. Names of fields are identified using small capital letters (e.g., NACA bit).

The names of procedure calls ~~Callable procedures~~ are identified by a name in bold type, such as **Execute Command** (see clause 5). Names of ~~procedural~~ arguments are denoted by capitalizing each word in the name. For instance, Sense Data is the name of an argument in the **Execute Command** procedure call.

Change 3 [Notation Conventions]: Modify 3.6.2 as follows:

Modify the subclause header and first paragraph as follows:

3.6.2 Notation for procedure calls ~~procedures and functions~~

In this standard, the model for functional interfaces between entities is a procedure call (see 3.1.x). ~~the callable procedure.~~ Such interfaces are specified using the following notation:

Modify the definitions of the arguments as follows:

Result: A single value representing the outcome of the procedure call ~~or function~~.

Procedure Name: A descriptive name for the function modeled by the procedure call ~~to be performed~~. When the procedure call model is used to describe a SCSI transport protocol service, the procedure name is the same as the service name.

Input-1, Input-2, …: A comma-separated list of names identifying caller-supplied input ~~data parameters~~ arguments.

Output-1, Output-2, …: A comma-separated list of names identifying output ~~data~~ arguments to be returned by the procedure call.

"[…]": Brackets enclosing optional or conditional ~~parameters and~~ arguments.

This notation allows ~~data parameters~~ arguments to be specified as inputs and outputs. The following is an example of a procedure call specification:

Modify the definition of the arguments in the example as follows:

Flag, if set to one, indicates that a matching item was located.

Pattern = … /* Definition of **Pattern** ~~parameter~~ argument */
~~Parameter~~ Argument containing the search pattern.

Item List = Item<NN> /* Definition of **Item List** as an array of NN **Item** ~~parameters~~ arguments*/

Item Found = Item … /* Item located by the search procedure call*/
This ~~parameter~~ argument is only returned if the search succeeds.

No other changes are required in this subclause.

Change 4 [distributed service model]: In 4.2 (The SCSI distributed service model), add boxes showing the initiator and target devices to figure 5 as shown here:

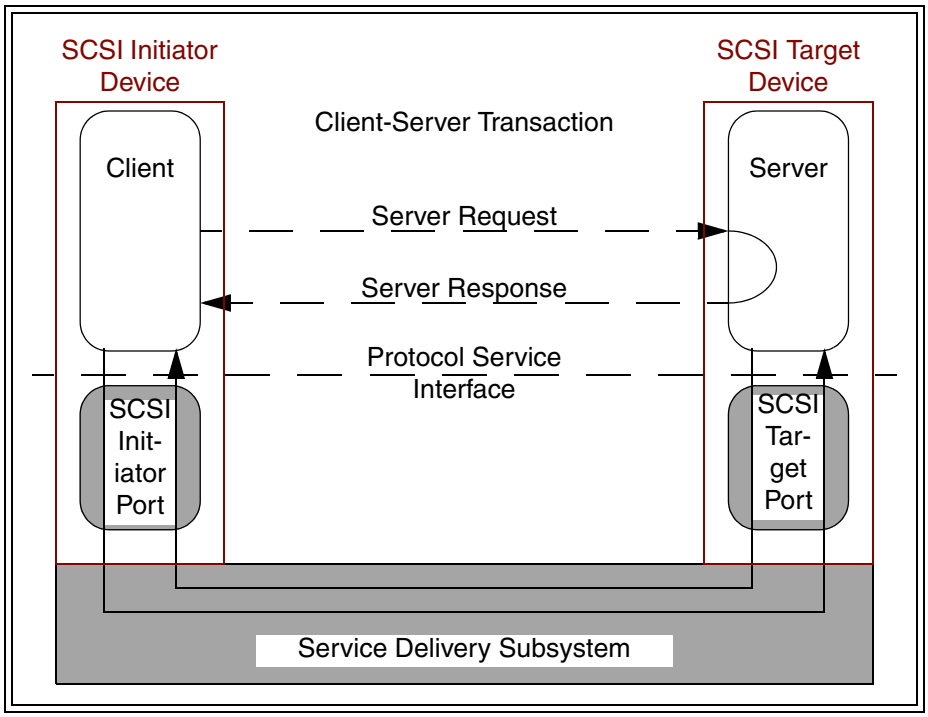

Figure 5 — Client-Server model

In 4.2 (The SCSI distributed service model), change the last paragraph from:

> Client-server relationships are not symmetrical. A client may only originate requests for service. A server may only respond to such requests. The client calls the server-resident procedure and waits for completion. From the client's point of view, the behavior of a remote service invoked in this manner is indistinguishable from a conventional procedure call. In this model, confirmation of successful request or response delivery by the sender is not required. The model assumes that delivery failures are detected by the client's SCSI port or within service delivery subsystem.

to:

> Client-server relationships are not symmetrical. A client may only originate requests for service. A server may only respond to such requests. The client requests an operation provided by a server located in another SCSI device and waits for completion, including transmission of the request and response to/from the remote server. From the client's point of view, the behavior of a service requested from a another SCSI device is indistinguishable from a request processed in the same SCSI device. In this model, confirmation of successful request or response delivery by the sender is not required. The model assumes that delivery failures are detected by the client's SCSI port or within service delivery subsystem.

Change 5 ['5.1 The Execute Command ~~remote~~ procedure']: Modify 5.1 as follows.

Modify the heading and first paragraph of 5.1 as follows:

5.1 The Execute Command ~~remote~~ procedure call

An application client requests the processing of a SCSI command by invoking the SCSI transport protocol services described in 5.4, the collective operation of which is conceptually modeled in the following ~~remote~~ procedure call:

Change 'status parameter' to 'Status argument' in the TASK COMPLETE and LINKED COMMAND COMPLETE **Service Response** descriptions.

Change 6 ['remote procedure call']: Change 'remote procedure call' to 'procedure call' in the following subclauses:

- 4.2 (The SCSI distributed service model)
- 4.6.1 (The service delivery subsystem object)
- 5.4.1 (SCSI transport protocol services in support of Execute Command Overview) [twice]
- 5.4.3.1 (Data transfer SCSI transport protocol services introduction)
- 5.8.1 (Unlinked command example)
- 5.8.2 (Linked command example) [twice]
- 6.3.2 (Hard reset)
- 6.3.4 (I_T nexus loss)
- 7.9 (Task management SCSI transport protocol services)

Change 7 ['call']: Change 'call' to 'procedure call' in the following subclauses:

- 4.15 (The SCSI model for distributed communications)
- 5.4.2 (Execute Command request/confirmation SCSI transport protocol service) [unless 03-002 is approved before this proposal]

Change 8 ['procedure']: Change 'procedure' to 'procedure call' in the following subclauses:

- 3.1.12 (code value)
- 5.1 (Execute Command…) [Data-In Buffer, Sense Data, Status, and SERVICE DELIVERY OR TARGET FAILURE]
- 5.8.1 (Unlinked command example) [twice]
- 5.8.2 (Linked command example) [twice]
- 7.9 (Task management SCSI transport protocol services) [3 times]
- 7.10 (Task management function example) [item 3]

Change 9 ['parameter']: Change 'parameter' to 'argument' in the following subclauses:

- 5.1 (Execute Command…) [Data-In Buffer, Sense Data, Status, and SERVICE DELIVERY OR TARGET FAILURE]
- 5.8.1 (Unlinked command example) [twice]
- 5.8.2 (Linked command example) [twice]
- 4.15 (The SCSI model for distributed communications) [3 times]
- 7.9 (Task management SCSI transport protocol services) [3 times]
- 7.10 (Task management function example) [item 3]

Change 10 ['data parameter']: Change 'data parameter' to 'argument' in the following subclauses:

- 3.1.12 (code value) [twice]
- 5.1 (Execute Command…) [Data-In Buffer, Sense Data, Status, and SERVICE DELIVERY OR TARGET FAILURE]
- 5.8.1 (Unlinked command example) [twice]
- 5.8.2 (Linked command example) [twice]
- 7.9 (Task management SCSI transport protocol services) [3 times]
- 7.10 (Task management function example) [item 3]

Change 11 ['Function call']: To both make the usage of 'procedure call' consistent and have a common nomenclature between clauses 5 and 7, change 'Function call:' to 'Request' in the following subclauses:

- 7.2 (ABORT TASK)
- 7.3 (ABORT TASK SET)
- 7.4 (CLEAR ACA)
- 7.5 (CLEAR TASK SET)
- 7.6 (LOGICAL UNIT RESET)
- 7.7 (QUERY TASK)
- 7.8 (WAKEUP)