Measuring Software Reuse in Object Oriented Systems and Ada Software

Santhi Karunanithi and James M. Bieman

Technical Report CS-93-125

October 13, 1993

Measuring Software Reuse in Object Oriented Systems and Ada Software

Santhi Karunanithi James M. Bieman

Department of Computer Science
Colorado State University
Fort Collins, Colorado 80523
bieman@cs.colostate.edu

Technical Report CS–93–125

October 1993

Abstract

The measurement of levels of software reuse is necessary to monitor improvements in software reuse. We develop a set of measurable reuse attributes appropriate to object oriented systems and a suite of metrics quantifying these attributes. Metrics suitable for the object based language Ada are identified. To measure the proposed reuse metrics a prototype Ada Reuse Metric Analyzer (ARMA) is implemented. The usefulness of ARMA is illustrated by measuring primitive reuse attributes for an Ada software system. Limitations of the tool and possible future directions are also discussed.

This report is adapted from the M.S. thesis, with the same title, by Santhi Karunanithi, 1993.

Address correspondence to J. Bieman, Department of Computer Science, Colorado State University, Fort Collins, CO 80523. (303)491-7096, Fax: (303) 491-6639, bieman@cs.colostate.edu

Research partially supported by NASA Langley Research Center, the Colorado Advanced Software Institute (CASI) and CTA, Inc. CASI is sponsored in part by the Colorado Advanced Technology Institute (CATI), an agency of the state of Colorado. CATI promotes advanced technology teaching and research at universities in Colorado for the purpose of economic development.

Contents

List of Figures

List of Tables

1 Introduction

Developing software in a predictable time period with acceptable quality and reliability is a recognized problem in the computer industry. There is a high demand for software systems of increasing complexity as well as the need to modify and extend existing software systems. Finding solutions to this "software crisis" is difficult [Bro87]. One important step towards solving this software crisis is via software reuse. One of the fundamental causes for the software bottleneck is that new software systems are usually developed from scratch even though much of the code in a given software system can be found in other similar systems [BP84, BP89, BR89]. Software reuse can potentially increase the quality and productivity of software development and maintenance. Software reuse reduces the amount of software that needs to be produced from scratch and thus allows a greater focus on quality. The reuse of well tested software should result in greater reliability and less testing time for new software. With reuse, software development becomes a capital investment. The measurement of reuse will help developers to monitor current levels of reuse and help provide insight in developing software that is easily reused. Though software reuse is an effective method for addressing the software crisis, software reuse does not occur without planning. In particular, programming languages can be designed to effectively support and promote software reuse. Object Oriented Programming Languages (OOPL) are one class of programming languages that have features to support reuse.

Though OOPLs have been designed to effectively support and promote software reuse, users need to understand the software engineering principles to produce programs that are easy to reuse. Users lacking a thorough familiarity with the principles of information hiding and data abstraction are unlikely to produce reusable programs. The measurement of reuse will help users to characterize the reusability of programs, monitor current levels of reuse and help provide insight in developing software that is easily reused.

Proponents assert that a major benefit of object oriented or object based design and programming is the generation of reusable software components [Mey87]. To support or refute claims that object oriented or object based software promotes software reuse, one must be able to measure reuse in these systems. Current reuse measures are not directed toward the object oriented approach. New definitions of attributes, abstractions and measures that support data abstraction, information hiding and inheritance constructs are needed to measure reuse in object oriented systems. The level of reuse must be measured in a way that provides information on a wide range of reuse attributes.

The objective of our work is to identify a suite of metrics for primitive reuse attributes appropriate to object oriented systems. The focus of reuse measurement is on internal product reuse measures. The attributes of reuse measurement are internal product attributes related to properties of particular software documents [Fen91]. Metric development follows the guidelines of scientific measurement principles as applied to software [BBF$^+$90, MGBB90, FM90]. In order to derive reuse measures, definitions of software reuse attributes, the documents to be measured, formal models of reuse attributes and a method for generating consistent numerical values from the documents and attributes need to be identified.

Reuse can not be completely quantified by a single reuse metric. *Primitive metrics* based on primitive reuse attributes are clearly components of the "level of reuse." *Composite metrics* are often less sensitive than the primitive metrics and must be defined carefully [Fen91]. Thus no attempt is made to combine the primitive components into meaningful composite metrics. The primitive components can later be used (or combined) to satisfy specific goals and questions of software developers. In this paper, we identify a set of measurable, intuitively meaningful reuse attributes in object oriented systems based on formal models and abstractions. The scope of the derived measures is restricted to static reuse measurements. In order to demonstrate the utility of the proposed metrics, a tool has to be developed to derive these metrics. A reuse measurement tool for measuring software reuse in Ada systems has been developed and empirical measures are obtained from an object based Ada system.

This paper is organized as follows. Chapter 2 presents the software measurement principles as applied to reuse and reviews the existing software reuse metrics for traditional software systems and object oriented software systems. The limitations of the surveyed research is the motivation for this paper. In Chapter 3, existing definitions of measurable reuse attributes appropriate to object oriented systems [Bie92] are extended with additional definitions of measurable attributes. Also, the documents to be measured and formal models or abstractions for capturing the reuse attributes are presented. Then reuse attributes and abstractions pertaining to object based language Ada are identified.

Chapter 4 describes the design of the reuse measurement tool, the *Ada Reuse Metric Analyzer* (ARMA), developed for measuring reuse in Ada systems. The measurement tool is an extension of the Anna tool set developed at Stanford [Men92]. Chapter 5 presents the data sets used in evaluating the reuse measurement tool, the derived reuse measures and an evaluation of the measures and the tool. Chapter 6 presents a summary of the results and general conclusions and implications of this work. Also, possible directions for future research are proposed.

2 Measuring Reuse : State of the Art

Though OOPLs have been designed to effectively support and promote software reuse, programmers need to apply software engineering principles to produce programs with maximum reuse. The measurement of reuse will help users to characterize the reusability of programs, monitor current levels of reuse and help provide insight in developing software that is easily reused. In this chapter existing software reuse metrics are reviewed and research needs are identified.

2.1 Software Measurement Theory

Any metric development should follow the guidelines of scientific measurement principles as applied to software [BBF$^+$90, MGBB90, FM90]. The attributes of reuse measurement are internal product attributes related to properties of particular software documents [Fen91]. Measurement theory suggest the following process [BBF$^+$90].

1. Identify and define intuitive and well-understood attributes of software reuse.

2. Specify precisely the documents and the attributes to be measured.

3. Determine formal models or abstractions which capture these attributes. Formal models are required to unambiguously produce numerical measurement values.

4. Derive mappings from the models of attributes to a number system. The mappings must be consistent with the orderings implied by the model attributes.

In order to derive reuse measures, definitions of software reuse attributes, the documents to be measured, formal models of reuse attributes and a method for generating consistent numerical values from the documents and attributes need to be identified.

2.2 Reuse Metrics for Traditional Software

Most of the current product reuse metrics are generally based on only one attribute namely program size or length. Conte [CDS86], Boehm [Boe81], Bailey [BB81], and Fenton [Fen91] describe reuse measures that are based on comparisons between the length or size of reused code and the size of newly written code in particular software products. Modifications to reused code are not considered by these researchers. Conte's reuse measure is for estimating coding effort. As reuse reduces coding effort, this affects effort estimation techniques. In a similar light, Boehm and Bailey use the size of reused code to adjust cost predictors.

Fenton proposed a measure of reuse based on the dependencies in an associated call graph. Fenton's reuse measures are *public reuse* and *private reuse*. Public reuse is defined as "the proportion of a product which was constructed externally." To use such a measure one must be able to clearly distinguish between the components that are from external sources and the components that are completely new. Private reuse is defined as the "extent to which modules within a product are reused within the same product." Using the call graph as an abstraction, Yin and Winchester measure the private reuse as $r(G) = e - n + 1$ where e is the number of edges and n is the number of nodes in the call graph [YW78]. Selby addresses the measurement of reuse with modifications [Sel89]. He classifies modules into categories based on the percentage of new versus reused code in a module. The categories are completely new modules, reused modules with major revisions ($\geq 25\% changed$), reused modules with slight revisions (< 25 % changed), and modules that are reused without change. Thus a count of the number of modules in each category provides a measure of reuse.

2.3 Reuse Metrics for Object Oriented Software

All the above metrics are leveled at conventional, non-object oriented software design and development. The object oriented approach is fundamentally different from the traditional approaches that emphasize a function oriented view that separated data and procedures. Software metrics developed with traditional methods in mind do not direct themselves to notions such as classes, inheritance, encapsulation and message passing.

Object Oriented Systems

Essentially, an object oriented software system is a collection of Abstract Data Types (ADTs) called classes. A *class* is an encapsulated specification of both the persistent state of an abstract data type and its operations or *methods*. An instantiation or *instance* of a class (a variable of the class type) is an *object*. A *message* is name of an operator and *message-passing* is a call of a method via a message. The only way to affect an object is by sending a message. OOPLs allow simultaneous existence of several methods for the same message. This is called *polymorphism*. *Dynamic binding of operator calls* in OOPL is run time resolution of method calls by messages. OOPLs support reuse through the following principles:

- *Information hiding* : separation of visible interface specifications and hidden bodies. Users can treat software units as black boxes to be manipulated via well defined interfaces supplied by the specification units. The user is completely immune to changes in the implementation units. Information hiding is supported via abstract data types and separate compilation.

- *Modularity* : property of a system that has been decomposed into a set of cohesive and loosely coupled modules.

- *Genericity* : operations whose behavior is largely indifferent to the types of their arguments are implemented by allowing the type to be a parameter of such functions. These function definitions are seen as templates which are instantiated when the type is supplied.

- *Inheritance* : a mechanism for building a new class by inheriting all the properties of an existing class (or classes) and by expanding (or changing), if desired, the functionality of the base class.

Object Based Systems

Object based languages support all the principles of OOPLs except inheritance. Ada is one such object based language. Ada is designed to support large scale development with reusable components. Ada supports software portability and reuse through the following features:

- *program units* : packages, subprograms, and tasks.

- *information hiding* : private clauses that allow separation of visible interface specifications and hidden bodies.

- *strong type checking* : compile time data type checking that improves the ability to reuse components without introducing context errors for data elements in the reused components.

- *generic program unit* : parameterized templates that allow the generation of software components.

- *program libraries* : separately compiled reusable program units.

The expected new version of Ada, Ada 9X [Coh90] incorporates additional features for generic parameters, overloading, and inheritance. A suite of new metrics especially designed to measure unique aspects of the object oriented approach is needed.

Reuse Metrics for Object Oriented Software

Chidamber et al. [CK91] propose a set of six metrics specifically developed for measuring elements contributing to size and complexity of object oriented design namely weighted methods per class, depth of inheritance tree, number of sub-classes subordinated to a class in the class hierarchy, coupling between objects, response for a class or cardinality of set of methods available to the object, and lack of cohesion in methods.

Gannon et al. [GKB86] show that a good background in software engineering principles is necessary to use the full capabilities of the Ada language. They propose three metrics for Ada packages: a component access metric and two package visibility metrics UA and PC. A component access is nothing but a reference to a data type. The component access metric is defined as the ratio of component accesses of objects with non-locally defined data types to lines of text in a program unit. The UA metric represents the ratio of units in which a package is accessed to those in which it could be made available by adding a **with** clause, given the current unit structure. The PC metric is defined as the ratio of the number of units in which a package must be visible to those in which it is visible.

Bieman defines classes of software reuse, identifies important perspectives of reuse, proposes relevant reuse abstractions, and suggests reuse attributes and associated metrics applicable to object oriented systems [Bie92]. He provides a framework for the derivation of software reuse measures. Reuse is classified along three axis: *public* or *private*, *verbatim* or *leveraged*, and *direct* or *indirect* reuse. Three perspectives of reuse are considered: the server perspective, the client perspective, and the system perspective.

2.4 Need for More Research

The need for software metrics designed to measure unique aspects of the object oriented approach is established. The level of reuse must be measured in a way that provides information on a wide range of reuse attributes. There is also a need for research on measuring reuse when the reused software is modified for new uses. There has been only minimal previous research on measuring reuse and even less research has been conducted on measuring reuse in object oriented systems. The results of Chidamber et al. [CK91] are very preliminary.

Gannon et al. claim that the component access metric measures the units resistance to changes. They also claim that the package visibility metric UA measures the perceived generality of the package. PC directly indicates the success of design decision to minimize visibility. The component access metric may not be applicable for object oriented programs written with private data types. However, the measures UA and PC are applicable for programs written in OOPLs as well as object based programs. The metrics are not exhaustive.

Bieman provides a framework for the derivation of reuse measures. This paper extends the work of Bieman by providing several additional reuse metrics. A set of measurable, intuitively meaningful reuse attributes is identified for object oriented systems. The reuse attributes and metrics are based on formal models and abstractions. All these metrics can be validated through empirical measurements. Hence a reuse measurement tool has to be developed to measure these metrics. In order to demonstrate the utility of these reuse metrics, a reuse measurement tool called *Ada Reuse Metric Analyzer* is developed and empirical measures are obtained from object based Ada systems.

3 Candidate Reuse Metrics

The level of reuse must be measured in a way that provides information on a wide range of reuse attributes. New definitions of attributes, abstractions and measures that support object oriented principles are needed to measure reuse in object oriented systems. Reuse can not be completely quantified by a single reuse metric. *Primitive metrics* based on primitive reuse attributes are clearly components of the "level of reuse." *Composite metrics* are often less sensitive than the primitive metrics and must be defined carefully [Fen91]. Thus no attempt is made to combine the primitive components into meaningful composite metrics. In this chapter, a set of measurable, intuitively meaningful reuse attributes in object oriented systems, documents to be measured, and formal models of reuse attributes are identified. The scope of measures is restricted to static reuse measurements.

3.1 Reuse Classification

Reuse can be measured along three axis: the *public* or *private*, classification as described by Fenton [Fen91], *verbatim* or *leveraged*, and *direct* or *indirect* as described by Bieman [Bie92]. *Verbatim reuse* is reuse without modifications. Reusing code from a library without change is verbatim reuse. Leveraged reuse is reuse with modifications. *Leveraged reuse* is accomplished when code from predefined classes is tailored to a new use. These modifications can be either *ad hoc* modifications (modifications not supported by the programming language) or more disciplined modifications (modifications made using language support.) Leveraged reuse is effectively measurable when it is supported by the language or programming environment. *Direct reuse* is reuse without going through an intermediate entity. *Indirect reuse* is reuse through an intermediate entity. When a module A uses another module B and a module C uses module A, then module C indirectly uses module B. The distance or level of indirection is the number of intermediate entities between a client (user) and a server (used). There may be different possible intermediate entities connecting a client and a server. So there may not be a unique distance of indirect reuse between two modules in a system.

Object Oriented Reuse

In OOPLs, verbatim reuse is accomplished by object instantiation of predefined classes, by the use of predefined classes as class components, and calls to subroutines. Object oriented support of leveraged reuse via genericity and inheritance provides an enhanced ability to analyze and measure leveraged reuse. Generics are simply templates for classes or subprograms. They are general versions of processes that can be modified by parameters at compilation time. Though reuse through genericity is verbatim in the sense that all the procedures are the same, it is also a leveraged reuse as the template is modified according to new generic parameter values. Hence for OOPLs reuse through genericity can be measured as a separate classification and reuse through inheritance can be measured as leveraged reuse. Thus, object oriented reuse can be classified as *verbatim reuse*, *generic reuse* and *leveraged reuse*.

Reuse can also be classified as direct or indirect. Since visible interfaces are separated from hidden bodies in object oriented systems, the amount of direct reuse in the interface of a class will increase the indirect reuse of that class. Hence, the amount of reuse in interfaces and hidden bodies need to be measured separately.

3.2 Reuse Perspectives

Different reuse attributes are visible when reuse is examined from different perspectives. Consider a system where individual modules access some set of existing software entities. A program unit being reused is considered a server and the unit accessing that program unit is considered a client. For example, when module M uses program unit S, M is a client and S is a server. Reuse can be observed from the perspectives of the server, the client, and the system. Each of these perspectives is relevant for the analysis and measurement of reuse in a system.

A set of potentially measurable attributes can be derived for OOPLs based on profiles of reuse from each perspective. In each perspective, reuse can be categorized as either verbatim, generic, or leveraged. As a result, we can define numerous measurable attributes [KB93]. Because of the numerous candidate reuse metrics that can be derived, they are presented here in a set of tables.

Client Perspective

The *client perspective* is the perspective of a new system or a new system component. The reuse analysis focuses on how a new class reuses existing library classes or other classes. A client can have verbatim reuse with a server class or global server method, can instantiate a generic server class, and/or inherit from a server class. Thus from a client perspective, measures of the number of servers reused, the number of times each server is reused, size of each server, etc. are important. Since an object or a method can be reused, the number of times a method in a server is invoked, and the size of that method are also relevant measures. The size of a reused component can be determined by source lines of code, number of bytes, function points and any relevant measure. Again, a client can reuse a server either directly or indirectly. Hence, the number of indirect

	Candidate Measures	Definition
1.	# Direct server classes	No. of classes included for verbatim usage.
2.	# Indirect server classes	No. of classes that direct servers have Using, Generic Instantiation, and Inheritance indirect relationships.
3.	# Server instance creations	No. of object instances of each server.
4.	# Server methods	No. of distinct server methods called by clients.
5.	# Server method instances	No. of calls to, or usages of each server method.
6.	# Global server methods	No. of non-generic global procedures/functions.
7.	# Global server instance	No. of usages of each global servers.
8.	Size of each server method	
9.	Size of server interface	
10.	Size of global definitions in server interface	
11.	Size of global definitions in server body	
12.	Size of each client method	
13.	Size of client interface	
14.	Size of global definitions in client interface	
15.	Size of global definitions in client body	
16.	# Paths to indirect servers	No. of paths connecting client and indirect servers.
17.	Length of paths to indirect servers	No. of edges in a path connecting client and indirect servers.

Table 1: Verbatim reuse measures from client perspective

servers, indirect clients, levels of indirection etc., can be measured. The level of indirection is the number of intermediate entities between a client and a server. This concept of "level" of indirection can be applied to all verbatim, leveraged and generic reuse. There may be different possible paths connecting a client and a server. Thus there may not be a unique distance of indirect reuse between a client and a server in a system. A count of number of paths between a particular client and a server gives a measure of indirect reuse. Table 1, Table 2, and Table 3 give reuse measures from the client perspective. Table 1 gives verbatim reuse measures, Table 2 gives generic reuse measures, and Table 3 gives leveraged reuse measures. These client perspective measures should be measured separately for the interface and body of the client.

Server Perspective

The *server perspective* is the perspective of the library or a particular library component. Given a class, the analysis focuses on how a server is being reused by all of the client classes in the system. A set of reuse measurements can be taken from the server perspective. These measurements can determine which library components are being reused and in what manner (verbatim, generic, leveraged, directly, indirectly). If the number of methods in a server class is larger, then the potential impact on children is larger since children inherit all the methods defined in the object. The number of children of a parent server class gives a measure of the potential influence a class has on the design. Thus, from the server perspective, relevant measures include the number of clients, the number of times the server is reused by all clients, size of each client, size of each client method invoking a server method of the server, the count of server methods, size of parameter list of server methods, and the number of times each method in the server is invoked. Under leveraged reuse classification, depth of inheritance tree, number of immediate sub-classes of a class in the class hierarchy are to be measured. A server class which is generic in nature can be reused through class instantiation or inheritance and a server class which is non-generic in nature can be reused through object instantiation or inheritance. Thus for a server class, there are only two classifications of reuse viz., verbatim and inheritance or generic and inheritance.

	Candidate Measures	Definition
1.	# Direct server classes	No. of generic classes included for generic instantiation.
2.	# Indirect server classes	No. of classes that direct servers have Using, Generic Instantiation, and Inheritance indirect relationships.
3.	# Server instance creations	No. of generic class instances of each generic server.
4.	# Object instance creations	No. of object instances of generic class instances.
5.	# Server methods	No. of distinct server methods called by clients.
6.	# Server method instances	No. of calls to, or usages of each server method.
7.	# Global server methods	No. of global generic procedures/functions.
8.	# Global server instances	No. of usages of each global servers.
9.	Size of each server method	
10.	Size of server interface	
11.	Size of global definitions in server interface	
12.	Size of global definitions in server body	
13.	Size of generic declarations in server	Size of generic parameters declaration.
14.	Size of each client method	
15.	Size of client interface	
16.	Size of global definitions in client interface	
17.	Size of global definitions in client body	
18.	# Paths to indirect servers	No. of paths connecting client and indirect servers.
19.	Length of paths to indirect servers	No. of relations or edges in a path connecting client and indirect servers.

Table 2: Generic reuse measures from client perspective

	Candidate Measures	Definition
	Leveraged	**Inheritance relationship measure**
1.	# Direct server classes	No. of direct superclasses.
2.	# Indirect server classes	No. of classes that direct servers have Using, Generic Instantiation and Inheritance indirect relationships.
3.	# Indirect parent servers	No. of indirect super classes for client.
4.	# Direct server methods inherited	No. of methods from server class available for client.
5.	# Direct server methods extended	No. of methods in client extended from corresponding server methods.
6.	# Direct server methods overridded	No. of methods in client overridding corresponding server methods.
7.	# Direct server overloaded methods	No. of methods in client overloading server methods.
8.	Size of each direct server method that is reused / extended	
9.	Size of direct server interface	
10.	Size of global definitions in server interface	
11.	Size of global definitions in server body	
12.	Size of each client method	
13.	Size of client interface	
14.	Size of global definitions in client interface	
15.	Size of global definitions in client body	
16.	Paths to indirect servers	No. of paths connecting client and indirect servers.
17.	Length of paths to indirect servers	No. of edges in a path connecting client and indirect servers.
18.	Paths to indirect parent servers	No. of paths connecting client and indirect parent servers.
19.	Length of paths to indirect parent servers	No. of edges in a path connecting client and indirect parent servers.

Table 3: Leveraged reuse measures from client perspective

Since a server can be reused directly or through an intermediate entity, the number of indirect clients, and levels of indirection are also relevant measures. Table 4 gives reuse measures from the server perspective under each reuse classification. In addition, the package visibility measures of Gannon et al. [GKB86] are also relevant measures from the server perspective. Hence the following measures are to be measured for each server:

Used: Number of units where information from the server is accessed or changed.

Current: Number of units where the server is currently visible.

Available: Number of units where the server could be made visible (for example by using *with* clause in an Ada system).

Proposed: Number of units where the server is visible given the current unit structure and server is made visible in lowest possible position (for example using *with* clauses in their lowest possible positions in Ada units).

System Perspective

The system perspective is a view of reuse in the overall system, including servers and clients. Included are both system-wide private reuse, and system-wide public reuse. The system perspective characterizes overall reuse of library classes in the new system. Measurable system reuse attributes include:

- Percentage of the new system source text imported from the library.
- Percentage of new system classes used verbatim from the library and percentage of those library classes that are imported.
- Percentage of new system classes derived from library templates and percentage of those library templates that are imported.
- Percentage of new system classes derived from library classes through inheritance and percentage of those library classes that are imported.
- The average number of verbatim, generic and leveraged clients for servers, and conversely, the average number of servers for clients.
- The average number of verbatim, generic and leveraged indirect clients for servers, and conversely, the average number of indirect servers for clients.
- The average length and number of paths between servers and clients for verbatim, generic and leveraged reuse.

3.3 Product Documents

Reuse attributes are internal product attributes related to properties of particular software documents. These software documents are either design documents or code documents. Design documents can be documents in the form of graph models such as call multigraph, inheritance hierarchy graph and Booch diagrams [Boo91]. Booch diagrams consist of four types of diagrams namely class diagrams, object diagrams, module diagrams and process diagrams. The class diagrams and object diagrams describe the existence and meaning of key abstractions that form the

	Candidate Measures	Definition
	Verbatim	**Using Relationship measure**
1.	# Direct clients	No. of classes having verbatim usage of server.
2.	# Indirect clients	No. of classes that have Using, Generic Instantiation, and Inheritance indirect relationships with direct clients.
3.	# Direct client invocations of server	No. of object instances in all clients.
4.	# Client invocations of server method	No. of calls to each server method in all clients.
5.	Count of server methods	No. of methods declared in server.
6.	Size of parameter list of each server method	
7.	Size of server interface	
8.	Size of each method in server	
9.	Size of global definitions in server interface	
10.	Size of global definitions in server body	
11.	# Paths to indirect clients	No. of paths connecting server and indirect clients.
12.	Lengths of paths to indirect clients	No. of edges in a path connecting server and indirect clients.
	Generic	**Generic Instantiation measure**
1.	# Direct clients	No. of client classes instantiating the generic server.
2.	# Indirect clients	No. of classes that have Using, Generic Instantiation, and Inheritance indirect relationships with direct clients.
3.	# Server instantiations	No. of class instances in all direct clients.
4.	# Direct client invocations of server	No. of object instances in all direct clients.
5.	# Client invocations of server method	No. of calls to each server method in all clients.
6.	Count of server methods	No. of methods declared in server.
7.	Size of parameter list of each server method	
8.	Size of server interface	
9.	Size of each method in server	
10.	Size of global definitions in server interface	
11.	Size of global definitions in server body	
12.	Size of generic declarations in server	Size of generic parameters declarations.
13.	# Paths to indirect clients	No. of paths connecting server and indirect clients.
14.	Lengths of paths to indirect clients	No. of edges in a path connecting server and indirect clients.
	Leveraged	**Inheritance Relationship measure**
1.	# Direct clients	No. of direct subclasses.
2.	# Indirect clients	No. of classes that have Using, Generic Instantiation, and Inheritance indirect relationships with direct clients.
3.	# Indirect child clients	No. of clients that have inherited indirectly from server.
4.	# Client invocations of server method	No. of times a method in server is reused in all its clients.
5.	Count of server methods	No. of methods declared in server.
6.	Size of parameter list of each server method	
7.	Size of server methods	
8.	Size of server interface	
9.	Size of global definitions in server interface	
10.	Size of global definitions in server body	
11.	Paths to indirect clients	No. of paths connecting server and clients.
12.	Length of paths to indirect clients	No. of edges in a path connecting client and server.
13.	Paths to indirect child clients	No. of paths connecting server and child clients.
14.	Length of paths to indirect child clients	No. of edges in a path connecting a server and child client.

Table 4: Reuse measures from server perspective

design. A *class diagram* shows the existence of classes and their relationships in the logical design of a system. A single class diagram represents all or part of the class structure of a system. The three important elements of a class structure are classes, class relationships, and class utilities (global modules). The class relationships include inheritance, using, and instantiation relationships. Using relationships indicate the use of predefined classes as class components and object instantiations of predefined classes. Instantiation relationships indicate the class instantiations of predefined generic classes. Inheritance relationships indicate the relations between sub-classes and super-classes. An object diagram is used to show the existence of objects and their relationships in the logical design of a system. The purpose of each object diagram is to illustrate the semantics of key mechanisms in the logical design. These graph models are abstractions that can capture the various reuse measures. These graph models can be derived from code documents also.

3.4 Reuse Tables

Reuse can not be measured by a single reuse measure. More specific attributes that contribute to the general notion of reuse have to be measured. The reuse measures proposed are primitive and are flexible enough to be used in various analyses. To support flexible use of the primitive metrics in future analyses, we specify the construction of a number of tables of primitive metrics derivable from these graph models. Though an incidence matrix will serve the purpose of measurement, these tables are proposed so that individual measures can be derived easily. These tables allow users to tailor their metric analysis to their own reuse goals. After creation of these tables, a user can use a spread sheet to analyze reuse.

Class Using Class Table (CUCT)

The CUCT table supports the measurement of verbatim reuse in classes from both the client and server perspectives. When a class A has verbatim use of another class B, A can instantiate B and access the methods of B. In the CUCT, each row represents a class in the measured system and each column represents a class that is used by a row class. Each table entry indicates both whether a row class 'uses' the associated column class, and the number of times a column class is instantiated in the row class. This CUCT table should be developed separately for interfaces of classes and hidden bodies of classes. By counting the number of entries in each row we can

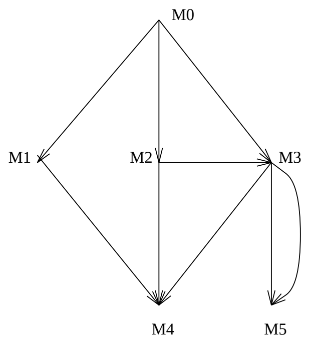


Figure 1: A call multigraph abstraction

compute the number of direct server classes for each row client. The sum of the count of instance creations for each row is the total number of server instance creations. The sum of entries for each column is the number of direct clients of a server class. The sum of the count of instances for each column is the number of direct client invocations of a server. The number of uses of

	M0		M1		M2		M3		M4		M5	
M0	0		1	1	1	1	1	1	0		0	
M1	0		0		0		0		1	1	0	
M2	0		0		0		1	1	1	1	0	
M3	0		0		0		0		1	1	1	2
M4	0		0		0		0		0		0	
M5	0		0		0		0		0		0	

Figure 2: CUCT table

each method of the server class can be obtained from the MMRT table which is described in Section 3.4. Figure 1 is an example call multigraph and Figure 2 is the corresponding CUCT table for that call multigraph.

Class Instantiating Generic Class Table (CIGCT)

The CIGCT table supports the measurement of generic reuse from both the client and server perspectives. When a class A instantiates a generic class B, A reuses B. In the CIGCT, each row represents a class in the measured system and each column represents a generic class. Each cell has three fields and the first field indicates if a row class instantiates a column class. The second field indicates the number of times a generic column class is instantiated in a row class. The third field indicates the number of times a generic object is instantiated.

The row sum of first field gives the number of direct generic server classes, the row sum of second field gives the number of server instance creations and the row sum of third field gives the number of object instance creations. The column sum of first field gives the number of direct clients for each generic server, the column sum of second field gives the the number of server instantiations in direct clients and the column sum of third field gives the number of client invocations of server. The number of use each method of the column class can be obtained from the MMRT table which is described in Section 3.4. As for CUCT table, the CIGCT table also must be developed separately for interfaces and hidden bodies of classes.

Method Method Reuse Table (MMRT)

The tables CUCT, and CIGCT indicate only class level verbatim and generic reuse. But a class method can reuse another class method and/or a global method. The table MMRT indicates the method level reuse from the client and server perspectives. In MMRT (except the first row and column), rows and columns are identified by method names. A method name can be a global procedure/function name or a procedure/function name in a class. Each cell in the first row indicates the size of the corresponding column methods and each cell in the first column indicates the size of the corresponding row methods. The methods of a class that are exported can be identified by the combination of class name, method name. Each table entry indicates both whether a row method calls a column method and the number of times a column method is called in the row method.

The first row of the MMRT indicates the size of server methods and the first column indicates the size of the column methods. This enables a user to measure the size of reused code in a new method. The count of the number of entries in each row is the number of server methods for each client method. The sum of the count of server instances for each row gives the total number of calls to server methods. The sum of entries for each column is the number of direct client methods. The sum of the count of server instances for each column is the number of direct client invocations of a server method.

Class Size Table (CST)

In addition to recording counts of various attributes, we also record the size of the software entities being examined. Thus, we need to measure the size of the reused and reusing code. In the MMRT table, we record the size of methods. In addition, we record the size of the packages or classes using the CST table. All rows in the CST are identified by class names. The table has four columns, one each for the size of class interface, the size of global definitions in the class interface, the size of global definitions in the class body, and the size of generic parameter declarations respectively. The last column is needed only for generic classes. Because visible interface specifications are separated from the hidden bodies in Ada systems, we separate these size measures.

Flexible Use of the Proposed Metrics Tables

The proposed tables support flexible use of the primitive metrics allowing users to tailor metric analysis for their specific needs. For a given client class, a user can determine the amount of verbatim reuse from the CUCT table, the amount of generic reuse from the CIGCT table, and the amount of both verbatim and generic method reuse from the MMRT table. Users can also determine the size of the reused code from the MMRT and CST tables. Similarly, a user can measure the reuse of a library class from the tables.

A first step in automated measurement of metrics analysis is the collection of data item counts. The second step is the calculation of metric scores through the formulation of the CUCT, CIGCT, MMRT and CST tables. The third step is the production of various reports based on obtained metrics. The form and content of these reports can be controlled by the user. The scope of deriving measures is limited to static reuse measurements.

3.5 Reuse Measurement in Ada

Ada is designed to support large scale development with reusable components. It supports software portability and reuse. The current version of Ada does not support inheritance. Hence all the measures defined for object oriented systems except leveraged reuse measures are applicable for Ada. However, the expected new version of Ada, Ada 9X [Coh90] incorporates more kinds of generic parameters, overloading, and inheritance thus making Ada an object oriented language. In this thesis, metrics are derived only for object based Ada systems. Hence, leveraged reuse metrics are not derived and empirically validated in this thesis. It is not possible to differentiate between the reuse of library classes from external systems, and reuse from within the new system. Ada does not provide any construct to differentiate between public library packages and local library packages. However in a particular programming environment the distinction between public reuse and private reuse can be made by those taking the measurements. In Ada, packages serve the purpose of classes in object oriented systems.

Reuse Table for Ada Systems

Ada allows nesting of packages and subprograms. The reuse tables given for object oriented systems have to be developed at each level of nesting. Also a package can have nesting of generic and non-generic packages. Hence, to support flexible use of primitive metrics in Ada systems, we propose an *Ada Reuse Table* (ART) combining all the reuse tables in a different format. This table supports both verbatim and generic reuse measurements from both the client and server perspectives. Each row in the table represents a package or a subprogram in the measured system and each column represents a library unit that is used by a row package. Each row and column unit will have its size information along with its name. Each table entry has four parts. The first part gives verbatim reuse count, the second part gives the generic instantiation count, the third part gives the generic object instantiation count and the fourth part gives the subprogram calls count. The sum of row entries of each part gives the total verbatim usage, total generic instantiation usage, total generic object instantiation usage and total subprogram calls respectively for that row client. Similarly, for a server, column sums give total verbatim and generic usage.

In order to demonstrate the utility of the proposed metrics, a reuse measurement tool *Ada Reuse Metric Analyzer* has been developed for measuring reuse in Ada systems. The next chapters describe the design of the ARMA and empirical measures that are obtained from an object based Ada system.

4 Ada Reuse Metric Analyzer

In order to facilitate the use of the software reuse metrics developed in Chapter 3, an automatable metric analysis tool is needed. This chapter presents the design of a prototype software tool, the "Ada Reuse Metric Analyzer" (ARMA), developed to measure reuse. The tool is designed to measure reuse properties in Ada programs. Instead of developing Ada syntactic and semantic analyzers from scratch, the Anna tool set developed at Stanford [Men92] is used as a base for the prototype tool design.

4.1 Anna Tool Set Description

Anna is a language extension of Ada which includes facilities for formally specifying the intended behavior of Ada programs [Luc90]. Anna is designed to augment Ada with precise machine-processable annotations so that formal methods of specification and documentation can be applied to Ada programs. The Program and Analysis Group at Stanford University has developed a prototype environment supporting the Anna language [Men92]. The tools implement a large subset of the Ada and Anna languages. The Anna-I tools are completely implemented in Ada and uses Descriptive Intermediate Attributed Notation for Ada (*DIANA*), [GWJB83] to form the intermediate representation of Ada programs. DIANA is an intermediate form of Ada programs which has been designed to be especially suitable for communication between the front and back ends of a compiler. DIANA encodes the results of lexical, syntactic, and static semantic analysis in the form of an Abstract Syntax Tree (AST). The Anna tool set includes the following tools:

- Intermediate Representation Toolkit (Extended DIANA Formal Interface and Implementation packages (AST), Ada/Anna Parser, Ada/Anna Pretty Printer, AST Disk to Memory package, and AST Memory to Disk package)
- Ada/Anna Static-Semantic Rules Checker
- Annotation Transformer
- Portable Ada/Anna Testing and Debugging System
- Ada/Anna AST Browser

The Intermediate Representation toolkit defines the common internal representation used by all the Anna-I tools. The Parser parses Anna and hence Ada text files into an AST representation. The Pretty Printer generates an ASCII text given an AST. The AST Disk to Memory and AST Memory to Disk package perform AST disk-to-memory conversions, and memory-to-disk conversions. The Ada/Anna Static-Semantic Checker Rules Checker inputs an AST and checks for the correctness of the static-semantics of the Ada and Anna code in the AST and updates the AST with semantic information. It works in both batch and incremental modes. The Annotation Transformer maps an Anna program to an equivalent Ada program. The AST Browser is an X-Windows based tool for graphically traversing and examining an Anna AST.

4.2 ARMA Design

The ultimate goal of the ARMA tool is to generate all possible reuse information. Though the current version of ARMA does not use all this information in providing reuse reports, ARMA is designed to be extended in future to do further analysis. There are three phases to measuring reuse using ARMA. ARMA consists of two parts - the Metric Generator and the Metric Analyzer. The Metric Generator extends the Anna tool set. Figure 3 shows the overall process used by ARMA.

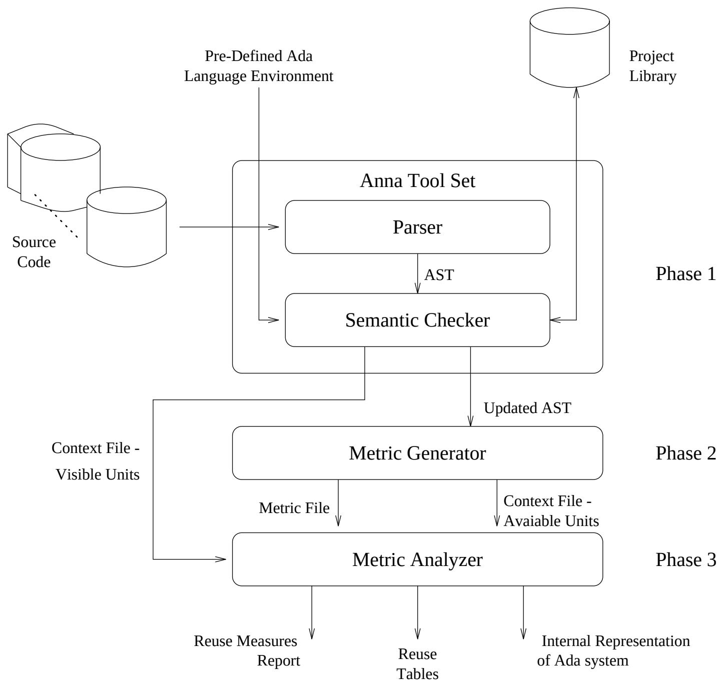


Figure 3: Ada Reuse Metric Analyzer (ARMA) Design

Phase 1

In the first phase, the Anna tool set is used to create an AST with semantic information. The parser reads the Ada source code and produces an AST. The semantic checker inputs this AST and information on the pre-defined language environment and then updates the AST with semantic information. Phase I is the only phase that uses the source code. The Ada make file is used to identify the compilation order of the units of an Ada application. The semantic checker keeps a project library to use in processing later compilation units. Two of the outputs of the semantic checker, the AST and the library context file created for each compilable unit, are used in the later processing. The library context file identifies all the library units that are visible to a particular compilable unit.

Phase 2

The Metric Generator of the ARMA inputs the updated AST and produces reuse measurements in an internal reuse data representation language. The Metric Generator consists of a *case* statement that recursively traverses the DIANA tree. The internal reuse data representation language allows the representation of all of the information necessary to generate reuse metrics, and identifies flow of control and flow of information throughout the system. For each compilable unit, ARMA creates a metric file in this internal language, and a library context file identifying all the library units that are made visible by a *with* clause. The BNF format of the internal reuse data representation language is given in Appendix A.

Phase 3

In the third phase, the Metric Analyzer of the ARMA reads all of the metric files and library context files of the compilable units using the compilation order specified in the Ada make file. ARMA forms an internal forest of trees in which the root of each tree corresponds to packages or subprograms in an Ada application. Nested packages and subprograms form the sub trees of each top tree node. Each root node can have two children corresponding to the interface and hidden body of that node. If there are nested packages and subprograms in the interface or body, they will form sub trees at one level below the root node or at the body node of the root node. As the nesting level increases, this tree nesting also increases. Corresponding to the sub trees in the interface, the body branch also will have sub trees. Each node will also maintain information about the library units that are visible, the library units that are made available by a *with* clause, server units information, and client units information. Figure 4 shows an example Ada system and its corresponding internal representation.

Once this internal formation of the forest is made, then all possible reuse metric information can be derived for that Ada application. After the creation of the internal forest of trees for an Ada system, ARMA forms the reuse table described in Section 3.5. At present the reuse table is formed only for units at the top level. Since Ada allows nesting of packages and subprograms in many levels, the reuse table can be created for each nesting level. The outputs of phase 3 are the internal forest representation, a reuse table and metrics report. Currently, only one form of a reuse metrics report is provided which contains reuse metrics from the client and server perspective for each unit at top level in the Ada system.

4.3 Limitations of Anna-I Tool Set

The Anna-I tool set is designed to run using the Verdix compiler. Significant time was required to import the tools from the Verdix environment to our Alsys environment. The Ada/Anna Static-Semantic Rules Checker has some severe limitations. In particular, we found several limitations to the semantic checker when used in ARMA. The checker does not check fully for the correctness of all Ada constructs, and hence the AST input to the Metric Generator of the ARMA does not have full semantic information. The checker fails to process deeply nested package or subprogram declarations. For our analysis, the major problems with the Anna tool is that generic unit instantiations and renaming declarations are not analyzed completely by the checker. The second

(a)
```
package X is
   subtype My_Int is Integer range 1..100;
   function Add (f : in My_Int; l : in Integer) return Integer;
end X;

package body X is
   function Add(f : in My_Int; l : Integer) return Integer is
   begin
      return (f+l);
   end Add;
end X;


With X;
package  Y is
   package Z is
      size : X.My_Int;
      function Compute return Integer;
   end Z;
   procedure Print;

end Y;


With Text_Io;
package body Y is
   package body Z is
      function Compute return Integer is
      begin
         return X.Add (1,2);
      end Compute;
   end Z;
   procedure Print is
   begin
      Text_Io.Put_Line ("An Example Ada System");
   end Print;
end Y;
```

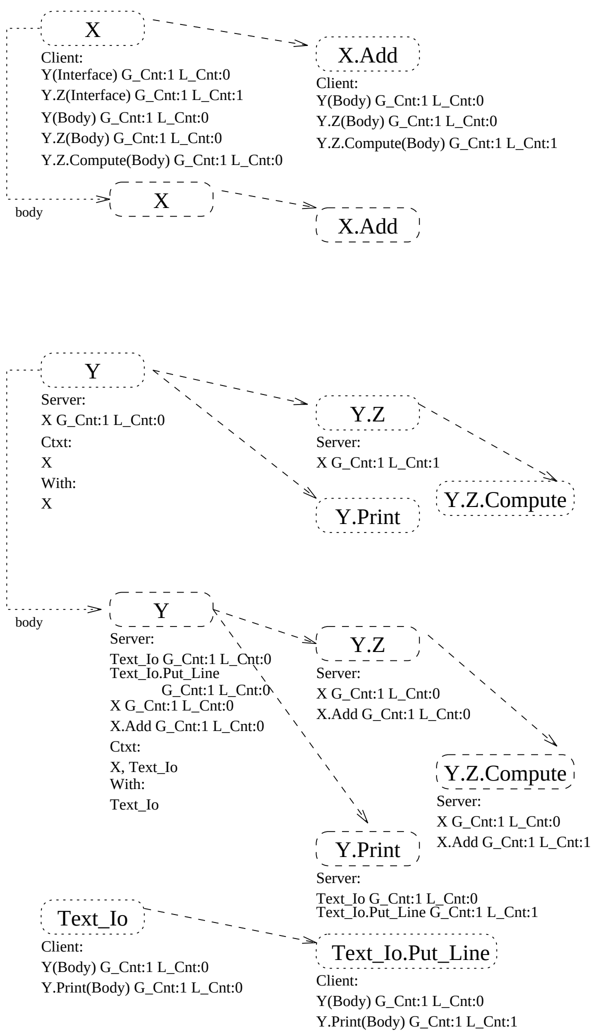
(b)


Figure 4: An Ada System (a) and its internal representation (b)

part of the ARMA resolves the problems in generic unit instantiations and renaming declarations by forming the internal representation of the Ada system.

5 Evaluating The Ada Reuse Metric Analyzer

In order to evaluate the Ada Reuse Metric Analyzer, it is necessary to have a proper set of test data. The data should provide measurement of reuse from both the client and server perspectives. ARMA was evaluated with two data sets provided by CTA, Inc.

5.1 The Test Data

The first data set contains 3 packages - *Global_Types*, *List_Manager*, and *Safe_Streams*. The Global_Types package contains the global data types declarations used in the data set. The List_Manager package contains a generic package *List_Type* providing the *list* ADT. The List_Type uses type declarations given in Global_Types. The Safe_Streams package is a collection of different stream data type packages and subprograms. It contains 10 subpackages and 2 subprograms. These subpackages and subprograms use the type information provided by the package Global_Types and the generic package List_Type. Figure 5 shows the units available in this data set. The size of the data set is 5715 source lines containing comments and 2935 lines without

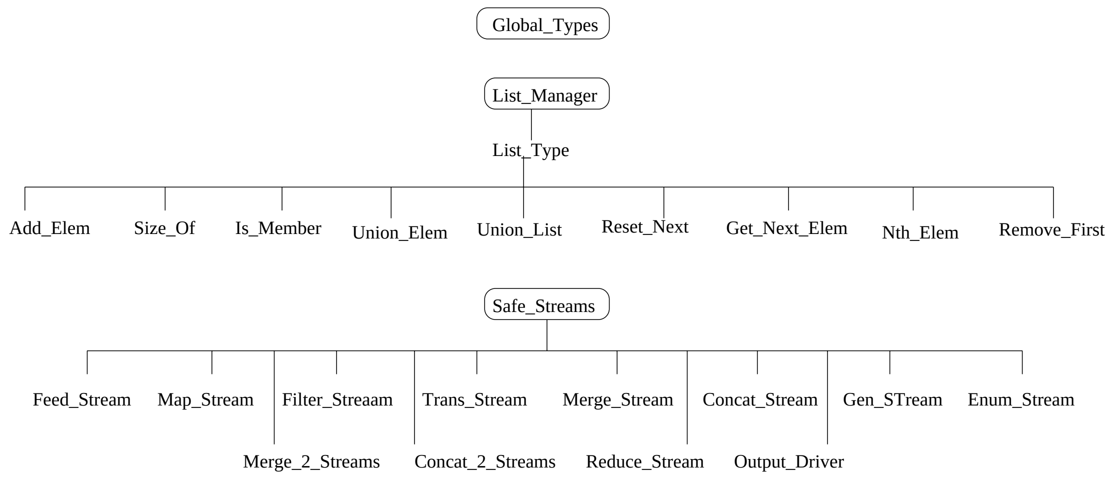


Figure 5: Data Set - 1

blank lines and comments (measured by not counting comments in the beginning of a line). These measures are count of lines in the Ada program files. ARMA measures LOC by counting number of semicolons. The size of the data set measured by the ARMA is 2199 LOC.

The second data set is a NASA system on POLAR/WIND Telemetry Simulator (POWITS). Much of the simulator design and code are reused from the Extreme Ultraviolet Explorer Telemetry Simulator (EUVETELS). The size of the data set is 67599 source lines containing comments and 36428 lines without blank lines and comments. The full data set has not yet been analyzed by the ARMA tool. Hence the reuse measurement results are given only for the first data set.

5.2 Results

The reuse measurement reports produced by ARMA for the data set are given in Appendices B and C. Appendix B contains the internal forest representation of the data set, and Appendix C contains the reuse report giving measurement at top level (root) of a tree in the forest. Figure 6 gives a portion of the internal forest representation corresponding to the package interface *Global_Types* and the package body *Safe_Streams*. The term *Glob_Size* refers to the size of library units import declaration. All the clients of a unit are listed under **Client**; all the servers of a unit are listed under **Server**; all the library units that are made visible to a unit are listed under **Ctxt**; and all library units that are imported using *with* clause are listed under **With**. There are two parts to the usage measurement namely **V G O C :: V G O C**. Each letter in this measurement refers to the following:

V Verbatim usage count,

G Generic instantiation count,

O Generic object instantiation count, and

C Number of calls made to the subprograms of the server.

```
* GLOBAL_TYPES (PAC_SPEC)                               Size: 5
  Client:
  LIST_MANAGER (PAC_SPEC)                               V  2 G 0 O 0 C 0 :: V 0 G 0 O 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                  V  2 G 0 O 0 C 0 :: V 0 G 0 O 0 C 0
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)            V  1 G 0 O 0 C 0 :: V 1 G 0 O 0 C 0
  LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_SPEC)           V  1 G 0 O 0 C 0 :: V 1 G 0 O 0 C 0
  LIST_MANAGER (PAC_BODY)                               V  1 G 0 O 0 C 0 :: V 0 G 0 O 0 C 0
  LIST_MANAGER.LIST_TYPE (PAC_BODY)                     V  1 G 0 O 0 C 0 :: V 1 G 0 O 0 C 0
  SAFE_STREAMS (PAC_SPEC)                               V  4 G 0 O 0 C 0 :: V 0 G 0 O 0 C 0
  SAFE_STREAMS.FEED_STREAM (GENERIC_PAC)                V  1 G 0 O 0 C 0 :: V 1 G 0 O 0 C 0
  SAFE_STREAMS (PAC_BODY)                               V 15 G 0 O 0 C 0 :: V 0 G 0 O 0 C 0
  SAFE_STREAMS.FEED_STREAM (PAC_BODY)                   V  2 G 0 O 0 C 0 :: V 1 G 0 O 0 C 0
  SAFE_STREAMS.MERGE_STREAM (PAC_BODY)                  V  7 G 0 O 0 C 0 :: V 1 G 0 O 0 C 0
  SAFE_STREAMS.GEN_STREAM (GENERIC_PAC)                 V  1 G 0 O 0 C 0 :: V 1 G 0 O 0 C 0
  SAFE_STREAMS.GEN_STREAM (PAC_BODY)                    V  2 G 0 O 0 C 0 :: V 1 G 0 O 0 C 0
  SAFE_STREAMS.MERGE_2_STREAMS (PAC_BODY)               V  2 G 0 O 0 C 0 :: V 1 G 0 O 0 C 0
  SAFE_STREAMS.REDUCE_STREAM (GENERIC_PROC)             V  1 G 0 O 0 C 0 :: V 1 G 0 O 0 C 0
  SAFE_STREAMS.OUTPUT_DRIVER (GENERIC_PROC)             V  1 G 0 O 0 C 0 :: V 1 G 0 O 0 C 0
  SAFE_STREAMS.OUTPUT_DRIVER (PROC_BODY)                V  1 G 0 O 0 C 0 :: V 1 G 0 O 0 C 0
  SAFE_STREAMS.REDUCE_STREAM (PROC_BODY)                V  1 G 0 O 0 C 0 :: V 1 G 0 O 0 C 0
  SAFE_STREAMS.MERGE_2_STREAMS.X_ADVANCE (PROC_BODY)    V  1 G 0 O 0 C 0 :: V 1 G 0 O 0 C 0
  SAFE_STREAMS.GEN_STREAM.CHECK_BUFFER (PROC_BODY)      V  1 G 0 O 0 C 0 :: V 1 G 0 O 0 C 0
  SAFE_STREAMS.MERGE_STREAM.BUMP (FUNC_BODY)            V  4 G 0 O 0 C 0 :: V 4 G 0 O 0 C 0
  SAFE_STREAMS.MERGE_STREAM.X_ADVANCE (PROC_BODY)       V  2 G 0 O 0 C 0 :: V 2 G 0 O 0 C 0
  SAFE_STREAMS.FEED_STREAM.CHECK_BUFFER (PROC_BODY)     V  1 G 0 O 0 C 0 :: V 1 G 0 O 0 C 0

* SAFE_STREAMS (PAC_BODY)                               Size: 149 Glob_Size: 3
  Server:
  GLOBAL_TYPES (PAC_SPEC)                               V 15 G  0 O  0 C   0 :: V 0 G 0 O 0 C 0
  LIST_MANAGER (PAC_SPEC)                               V  0 G 14 O 24 C 128 :: V 0 G 0 O 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                  V  0 G 14 O 24 C 128 :: V 0 G 0 O 0 C 0
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)            V  0 G  0 O  0 C  37 :: V 0 G 0 O 0 C 0
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)       V  0 G  0 O  0 C  25 :: V 0 G 0 O 0 C 0
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)           V  0 G  0 O  0 C  25 :: V 0 G 0 O 0 C 0
  LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)         V  0 G  0 O  0 C  13 :: V 0 G 0 O 0 C 0
  LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)      V  0 G  0 O  0 C  13 :: V 0 G 0 O 0 C 0
  LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_SPEC)           V  0 G  0 O  0 C  15 :: V 0 G 0 O 0 C 0
  DIRECT_IO (GENERIC_PAC)                               V  0 G  1 O  2 C   3 :: V 0 G 0 O 0 C 0
  DIRECT_IO.READ (PROC_SPEC)                            V  0 G  0 O  0 C   1 :: V 0 G 0 O 0 C 0
  DIRECT_IO.CLOSE (PROC_SPEC)                           V  0 G  0 O  0 C   1 :: V 0 G 0 O 0 C 0
  DIRECT_IO.DELETE (PROC_SPEC)                          V  0 G  0 O  0 C   1 :: V 0 G 0 O 0 C 0
  Ctxt:
  GLOBAL_TYPES
  DIRECT_IO
  LIST_MANAGER
  With:
  LIST_MANAGER
  DIRECT_IO
```

Figure 6: A Portion of Internal Forest Representation

The first part **V G O C** gives the value of direct or indirect usage and the second part gives only the direct usage value. For example, in Figure 6, *Safe_Streams.Merge_Stream (package body)* has verbatim use of *Global_Types* directly once and indirectly 6 times. Of the 6 indirect usage counts 4 of them are from the subprogram *Safe_Streams.Merge_Stream.Bump* and 2 are from *Safe_Streams.Merge_Stream.X_Advance.*

Table 5 gives the summary of reuse measurements in terms of *Ada Reuse Table* described in Section 3.5. Since Ada allows nesting of packages and subprograms, the reuse measurements can be measured at each level of nesting. The Table 6 gives the reuse measurements for the nested packages and subprograms at first level. In these tables, the reuse measurements for Ada standard packages are not included though they are given in the internal forest representation. The measurements give an indication of the amount of reuse within each unit and also the amount of usage of each library unit. The tables can be created for each level of nesting and reuse at any level can be analyzed separately. The results show that the generic package *List_Manager.List_Type* is used extensively in the package *Safe_Streams*. This is expected since *list* ADT is a general purpose ADT that is used in many applications. The zero entries for Safe_Streams package interface indicate that the List_Manager is used only in the hidden body of Safe_Streams and hence its use is not exported through the interface packages or operations of Safe_Streams. A person using Safe_Streams does not have to know about List_Manager usage in Safe_Streams. From the internal forest representation given in Appendix B we find that the List_Manager package is made visible to Safe_Streams (package body) by a "with" clause. Hence List_Manager is also visible to all the subunits of Safe_Streams (package body) though it is not used by all these subunits. From the Table 6 we find that the subunits Safe_Streams.Enum_Stream (body), Safe_Streams.Reduce_Stream (body) and Safe_Streams.Output_Driver (body) do not use List_Manager though they have visibility to List_Manager. Hence we get the following values for Gannon et al. measures [GKB86] corresponding to List_Manager package.

Used: 9 units

Current: 13 units

Available: 14 units

Proposed: 9 units

UA(List_Manager) or perceived generality of List_Manager is calculated as **Used/Available** = 0.64 and the PC(List_Manager) or excess visibility of List_Manager is calculated as **Proposed/Current** = 0.69. Since UA(List_Manager) is not very low, we conclude that it is a useful package. The PC(List_Manager) indicates the quality of the system structure rather than the reusability of List_Manager.

Additional data sets need to be processed by ARMA to further evaluate the proposed metrics and derive new metrics. The main objective of the ARMA design is to generate all possible reuse information. The results of the Metric Analyzer can be used as input to another *report generator* phase to prepare different reports. ARMA has been designed to be extended for additional analyses.

5.3 Limitations of Anna Tool Set

In the process of testing the ARMA, we found several limitations to the Anna tool set in addition to the limitations already described. The test data has to be modified (without changing its reuse characteristics) for processing by Anna semantic checker. For example, some type declarations

Client		**Server**								**Total**			
		Global_Types Size: 5				*List_Manager* Size: 23							
	Size	V	G	O	C	V	G	O	C	V	G	O	C
Global_Types(Interface)	5	0	0	0	0	0	0	0	0	0	0	0	0
Global_Types(Body)	2	0	0	0	0	0	0	0	0	0	0	0	0
List_Manager(Interface)	23	2	0	0	0	0	0	0	0	2	0	0	0
List_Manager(Body)	90	1	0	0	0	0	0	0	0	1	0	0	0
Safe_Streams(Interface)	290	4	0	0	0	0	0	0	0	4	0	0	0
Safe_Streams(Body)	1789	15	0	0	0	0	14	24	128	15	14	24	128
Total		22	0	0	0	0	14	24	128	22	14	24	128

Table 5: Summary of Results

in the body of *Safe_Streams.Feed_Stream* were moved to the private part of the corresponding interface. The Anna semantic checker cannot process nested packages correctly. Some of the Anna errors can be ignored while many others require some program manipulation. As a result, testing ARMA is time consuming since the ARMA depends on the output of semantic checker. Note that the aim of this research is to develop a prototype tool to evaluate our ideas concerning reuse measurement. ARMA is not intended to be a production tool.

6 Conclusions

Though object oriented languages have been designed to support and promote reuse, programmers need to use proper design methods to produce programs with maximum reuse. Tools that measure attributes related to reuse in software systems can identify properties that make software more reusable. Such empirical results can help determine the amount of reuse in existing systems and identify the most frequently reused software components. Reuse measurement will also help users gain insights to help develop software that is easily reused.

6.1 Reuse Metrics

In this paper we review existing software reuse metrics, existing reuse metrics more appropriate to object oriented systems, and identify the limitations of the metrics. We propose additional software reuse metrics appropriate to object oriented systems. We identify different reuse attributes pertaining to object oriented languages, and present the documents to be measured and formal models or abstractions for capturing the reuse attributes.

The reuse metrics can be classified along three dimensions: *public* or *private*, *verbatim* or *generic* or *leveraged*, and *direct* or *indirect* reuse. Public reuse is reuse of externally constructed software and private reuse is reuse of software within a application product. Verbatim reuse is reuse without modifications. Generic reuse is reuse of generic templates and leveraged reuse is reuse with modifications supported through inheritance. Direct reuse is reuse without going through an intermediate entity and indirect reuse is reuse through an intermediate entity. Also reuse can be measured from different perspectives. It can be measured from the perspective of the client or the server or from the system. A number of potentially measurable attributes can be derived based on these perspectives. Some of them are based on simple counts such as the number of direct or indirect clients or servers, the number of server class instances, the number of object instances, the number of calls to a method in the server, the number of library units that are visible to a unit, and the number of library units imported explicitly in a unit.

Client		**Server**								**Total**			
		Global_Types Size: 5				*List_Manager* Size: 23							
	Size	V	G	O	C	V	G	O	C	V	G	O	C
List_Manager. List_Type(Gen.Int)	19	2	0	0	0	0	0	0	0	2	0	0	0
List_Manager. List_Type(Body)	89	1	0	0	0	0	0	0	0	1	0	0	0
Safe_Streams. Feed_Stream(Gen.Int)	23	1	0	0	0	0	0	0	0	1	0	0	0
Safe_Streams. Map_Stream(Gen.Int)	25	0	0	0	0	0	0	0	0	0	0	0	0
Safe_Streams. Filter_Stream(Gen.Int)	24	0	0	0	0	0	0	0	0	0	0	0	0
Safe_Streams. Merge_Stream(Gen.Int)	23	0	0	0	0	0	0	0	0	0	0	0	0
Safe_Streams. Concat_Stream(Gen.Int)	23	0	0	0	0	0	0	0	0	0	0	0	0
Safe_Streams. Trans_Stream(Gen.Int)	27	0	0	0	0	0	0	0	0	0	0	0	0
Safe_Streams. Gen_Stream(Gen.Int)	17	1	0	0	0	0	0	0	0	1	0	0	0
Safe_Streams. Enum_Stream(Gen.Int)	26	0	0	0	0	0	0	0	0	0	0	0	0
Safe_Streams. Merge_2_Streams(Gen.Int)	33	0	0	0	0	0	0	0	0	0	0	0	0
Safe_Streams. Concat_2_Streams(Gen.Int)	33	0	0	0	0	0	0	0	0	0	0	0	0
Safe_Streams. Reduce_Stream(Gen.Int)	16	1	0	0	0	0	0	0	0	1	0	0	0
Safe_Streams. Output_Driver(Gen.Int)	16	1	0	0	0	0	0	0	0	1	0	0	0
Safe_Streams. Feed_Stream(Body)	268	2	0	0	0	0	2	3	21	2	2	3	21
Safe_Streams. Map_Stream(Body)	120	0	0	0	0	0	1	2	6	0	1	2	6
Safe_Streams. Filter_Stream(Body)	124	0	0	0	0	0	1	2	6	0	1	2	6
Safe_Streams. Marge_Stream(Body)	181	0	0	0	0	0	3	4	26	0	3	4	26
Safe_Streams. Concat_Stream(Body)	168	0	0	0	0	0	2	3	21	0	2	3	21
Safe_Streams. Trans_Stream(Body)	146	0	0	0	0	0	1	2	10	0	1	2	10
Safe_Streams. Gen_Stream(Body)	177	0	0	0	0	0	2	4	20	0	2	4	20
Safe_Streams. Enum_Stream(Body)	94	0	0	0	0	0	0	0	0	0	0	0	0
Safe_Streams. Merge_2_Streams(Body)	157	0	0	0	0	0	1	2	6	0	1	2	6
Safe_Streams. Concat_2_Streams(Body)	172	0	0	0	0	0	1	2	12	0	1	2	12
Safe_Streams. Reduce_Stream(Body)	16	1	0	0	0	0	0	0	0	1	0	0	0
Safe_Streams. Output_Driver(Body)	17	1	0	0	0	0	0	0	0	1	0	0	0
Total		22	0	0	0	0	14	24	128	22	14	24	128

Table 6: Summary of Results with one level nesting

6.2 Measurement Tool

To illustrate the utility of the proposed metrics, a measurement tool is designed to take these reuse metrics. We identify the reuse attributes pertaining to the object based language Ada and present the design of a reuse measurement tool for Ada, *ARMA*, and describe all the reuse measurements that are derived from the tool. The prototype tool ARMA has been implemented and initial empirical evaluation of the measurement tool was accomplished by measuring proposed reuse metrics for a selected software supplied by CTA, Inc.

The ARMA tool extends the Anna-I tool set developed at Stanford [Men92]. There are two parts to the ARMA tool, Metric Generator and Metric Analyzer. The Metric Generator takes the AST with semantic information as input and generates reuse information using an internal reuse representation language. The Metric Analyzer combines the metric information for all compilable units and forms an internal forest representation of the Ada system and the Ada Reuse Table. From this internal forest representation various reuse reports can be prepared. At present the reuse table is created only for the top level units in an Ada system.

6.3 Future Directions

Results from the tool testing provided initial profiles of software reuse. Our initial measurements also provide insight into what needs to be measured in future. Some of the insights are as follows:

- Having a single data type per package visible rather than more than one will increase the reusability of that package. This may also decrease the complexity of that Ada system. Distributing representation information of an object using more than one data type rather than centralizing it in a single data type will make programs more difficult to change. Changes in representation could involve changes in many compilation units.

- Too many levels of nesting of units may affect reuse since one has to understand the nesting levels of units to reuse some nested unit.

The measures obtained in this paper are limited and measure only primitive properties of reuse metrics. In this paper we have not made any attempt to analyze the measures to combined them to form composite metrics. Thus, one possible extension of this work will be to analyze primitive reuse metrics to produce composite metrics that can be closely tied to actual reuse and *reusability* in object oriented systems.

To further validate the ARMA tool and the reuse measures proposed in this paper, we need to measure reuse using more test data. For example, the NASA system on POLAR/WIND Telemetry Simulator can be a potential test data. Furthermore, the ARMA tool need to be extended to provide many reuse reports that highlight different reuse aspects.

References

[BB81] J. Bailey and V. Basili. A meta-model for software development resource expenditures. In *Proc. 5th Int. Conf. Software Engineering (ICSE-5)*, pages 107–116, 1981.

[BBF+90] A. Baker, J. Bieman, N. Fenton, A. Melton, and R. Whitty. A philosophy for software measurement. *Journal of Systems and Software*, 12(3):277–281, July 1990.

[Bie92] J. Bieman. Deriving measures of software reuse in object-oriented systems. proc. bcs workshop on formal aspects of measurement (1991). In T. Denvir, R. Herman, and

R. Whitty, editors, *Formal Aspects of Measurement*, pages 63–83. Springer-Verlag, 1992.

[Boe81] B. Boehm. *Software Engineering Economics.* Prentice-Hall, Englewood Cliffs, NJ, 1981.

[Boo91] G. Booch. *Object oriented design with applications.* The Benjamin/Cummings Publishing, Inc., Menlo Park, CA, 1991.

[BP84] T. Biggerstaff and A. Perlis, editors. Special Issue on software reuseability. *IEEE Trans. Software Engineering*, volume SE-10(5). 1984.

[BP89] T. Biggerstaff and A. Perlis, editors. *Software Reusability Vols. I, II.* ACM Press. Addison-Wesley, 1989.

[BR89] T. Biggerstaff and C. Richer. Reusability framework, assessment, and directions. In T. Biggerstaff and A. Perlis, editors, *Software Reusability Vols. I, II.* ACM Press. Addison-Wesley, 1989.

[Bro87] F. Brooks. No silver bullet: Essence and accidents of software engineering. *IEEE Computer*, 20(4):10–19, April 1987.

[CDS86] S. Conte, H. Dunsmore, and V. Shen. *Software Engineering Metrics and Models.* The Benjamin/Cummings Publishing, Inc., Menlo Park, CA, 1986.

[CK91] S. Chidambar and C. Kemerer. Towards a metrics suite for object oriented design. In *Proc. OOPSLA*, pages 197–211, 1991.

[Coh90] S. Cohen. Ada 9x project report: Ada support for software reuse. Technical Report SEI-90-SR-16, Software Engineering Institute, Carnegie Mellon University, Pittsburg, PA, Oct 1990.

[Fen91] N. Fenton. *Software Metrics A Rigorous Approach.* Chapman & Hall, London, 1991.

[FM90] N. Fenton and A. Melton. Deriving structurally based software measures. *Journal of Systems and Software*, 12(3):177–187, July 1990.

[GKB86] J. Gannon, E. Katz, and V. Basili. Metrics for ada packages : An initial study. *Communications of the ACM*, 29(7):616–623, 1986.

[GWJB83] G. Goos, W. Wulf, A. Evans Jr., and K. Butler. *DIANA, An Intermediate Language for Ada*, volume 161. Springer-Verlag, 1983.

[KB93] S. Karunanithi and J. Bieman. Candidate reuse metrics for object oriented and ada software. *Proc. IEEE-CS Int. Software Metrics Symposium*, pages 120–128, May 1993.

[Luc90] David C. Luckham, editor. *Programming with Specifications: An Introduction to Anna, A language for specifying Ada programs.* Springer-Verlag, 1990.

[Men92] G. Mendal. The anna-i user's guide and installation manual, version 1.4 edition. Technical Report ERL 456, Stanford University, Computer Systems Lab, Stanford, CA, Sept 1992.

[Mey87] B. Meyer. Reusability: The case for object oriented design. *IEEE Software*, 4(2):50–64, Mar 1987.

[MGBB90] A. Melton, D. Gustafson, J. Bieman, and A. Baker. A mathematical perspective for software measures research. *IEE Software Engineering Journal*, 5(5):246–254, 1990.

[Sel89] R. Selby. Quantitative studies of software reuse. In Ted J. Biggerstaff and Alan J. Perlis, editors, *Software Reusability Vol. II Applications and Experiences*, pages 213–233. Addison-Wesley, 1989.

[YW78] B. Yin and J. Winchester. The establishment and use of measures to evaluate the quality of system designs. In *Proc. Software Quality and Assurance Workshop*, pages 45–52, 1978.

A BNF for Internal Reuse Data Representation Language

```
compilation_unit           ::= library_unit | secondary_unit
library_unit               ::= subprogram_declaration | package_declaration | generic_declaration |
                               generic_instantiation | subprogram_body
secondary_unit             ::= library_unit_body | subunit
library_unit_body          ::= subprogram_body | package_body
subunit                    ::= Subunit parent_unit_name < cr > proper_body
proper_body                ::= subprogram_body | package_body | task_body
subprogram_declaration     ::= proc_declaration | func_declaration
proc_declaration           ::= proc_specification
                               end_proc_specification
proc_specification         ::= proc_name < cr > [formal_part] [import_part]
end_proc_specification     ::= End_Of proc_name size_declaration
func_declaration           ::= func_specification
                               end_func_specification
func_specification         ::= func_name < cr > [formal_part]
                               result_formal_part [import_part]
end_func_specification     ::= End_Of func_name size_declaration
proc_name                  ::= PROC_SPEC unit_name
func_name                  ::= FUNC_SPEC unit_name
formal_part                ::= par_specification [par_specification]
par_specification          ::= mode par_name:type_name=boolean < cr >
mode                       ::= Mode_In | Mode_Out | Mode_In_Out
boolean                    ::= Y | N
result_formal_part         ::= Res type_name < cr >
size_declaration           ::= Size value < cr >
import_part                ::= import_par_specification [import_par_specification]
import_par_specification   ::= Par_Imp type_name < cr >
package_declaration        ::= package_specification
                               end_package_specification
package_specification      ::= pac_name < cr >
                               [basic_declaration]
                               [Priv < cr > [basic_declaration]
                                End_Of Priv size_declaration]
end_package_specification  ::= End_Of pac_name size_declaration
pac_name                   ::= PAC_SPEC package_name
package_body               ::= pac_body_name < cr >
                               [basic_declaration]
                               body_detail
                               End_Of pac_body_name size_declaration
pac_body_name              ::= PAC_BODY package_name
generic_declaration        ::= Gen_Type generic_specification
generic_specification      ::= proc_specification
                               generic_par_list
                               End_Of Gen_Type proc_name size_declaration |
                               func_specification
                               generic_par_list
                               End_Of Gen_Type func_name size_declaration |
                               package_specification
                               generic_par_list
                               End_Of Gen_Type pac_name size_declaration
generic_par_list           ::= Generic_par < cr >
                               gen_par_list [gen_par_list]
                               End_Of Generic_Par size_declaration
                               [generic_par_import]
gen_par_list               ::= gen_mode | type_use | subprogram_declaration
gen_mode                   ::  Gen_Mode_In type_name < cr >
generic_par_import         ::= Gen_Par_Imp type_name < cr > [generic_par_import]
generic_instantiation      ::= package_instantiation |
                               proc_instantiation |
                               func_instantiation
rename_use                 ::= package_rename |
                               proc_rename |
                               func_rename
package_instantiation      ::= package_specification
                               Gen_Ins package_name < cr >
                               end_package_specification
```

```
proc_instantiation       ::= proc_specification
                             Gen_Ins unit_name < cr >
                             end_proc_specification
func_instantiation       ::= func_specification
                             Gen_Ins unit_name < cr >
                             end_func_specification
package_rename           ::= package_specification
                             Rena package_name < cr >
                             end_package_specification
proc_rename              ::= proc_specification
                             Rena unit_name < cr >
                             end_proc_specification
func_rename              ::= func_specification
                             Rena unit_name < cr >
subprogram_body          ::= proc_body_declaration |
                             func_body_declaration
proc_body_declaration    ::= proc_body_name < cr > [formal_part]
                             [import_part] [basic_declaration]
                             body_detail
                             End_Of proc_body_name size_declaration
func_body_declaration    ::= func_body_name < cr > [formal_part]
                             result_formal_part [import_part]
                             [basic_declaration] [body_detail]
                             End_Of func_body_name size_declaration
proc_body_name           ::= PROC_BODY unit_name
func_body_name           ::= FUNC_BODY unit_name
body_detail              ::= [usage_statements] [body_detail]
basic_declaration        ::= declarative_part [declarative_part]
declarative_part         ::= object_use |
                             type_use |
                             constant_use |
                             rename_use |
                             subprogram_declaration |
                             package_declaration |
                             task_declaration |
                             generic_declaration |
                             generic_instantiation |
                             body
object_use               ::= Var type_name < cr >
type_use                 ::= Typ type_name < cr >
constant_use             ::= Con type_name < cr >
usage_statements         ::= call_statement |
                             use_statement
call_statement           ::= Call unit_name boolean < cr >
                             [act_par_list] [usage_statement]
                             End_Of Call < cr >
act_par_list             ::= General_Ass mode:type_name < cr >
                             [act_par_list]
use_statement            ::= Usage type_name < cr >
body                     ::= proper_body |
                             body_stub
body_stub                ::= Stub unit_declaration
unit_declaration         ::= subprogram_body_declaration |
                             package_body_declaration |
                             task_body_declaration
task_declaration         ::= task_specification
task_specification       ::= TASK_TYP task_name < cr >
                             [entry_declaration]
                             End_Of TASK_TYP task_name size_declaration
entry_declaration        ::= ENTRY_SPEC entry_name < cr >
                             [formal_part] [import_part]
                             End_Of ENTRY_SPEC entry_name size_declaration
task_body                ::= TASK_BODY task_name < cr >
                             [basic_declaration]
                             body_detail
                             End_Of TASK_BODY task_name size_declaration
```

B Internal Forest Representation For Data Set - 1

```
* IO_EXCEPTIONS (PAC_SPEC)                          Size: 9
  Client:
  SEQUENTIAL_IO (GENERIC_PAC)                       V 7 G 0 0 0 C 0 :: V 7 G 0 0 0 C 0
  DIRECT_IO (GENERIC_PAC)                           V 7 G 0 0 0 C 0 :: V 7 G 0 0 0 C 0
  TEXT_IO (PAC_SPEC)                                V 8 G 0 0 0 C 0 :: V 8 G 0 0 0 C 0
* SEQUENTIAL_IO (GENERIC_PAC)                       Size: 36 Glob_Size: 1 Gen_Size: 1
  Server:
  IO_EXCEPTIONS (PAC_SPEC)                          V 7 G 0 0 0 C 0 :: V 7 G 0 0 0 C 0
  Ctxt:
  IO_EXCEPTIONS
  With:
  IO_EXCEPTIONS
    * SEQUENTIAL_IO.CREATE (PROC_SPEC)              Size: 4
    * SEQUENTIAL_IO.OPEN (PROC_SPEC)                Size: 4
    * SEQUENTIAL_IO.CLOSE (PROC_SPEC)               Size: 1
    * SEQUENTIAL_IO.DELETE (PROC_SPEC)              Size: 1
    * SEQUENTIAL_IO.RESET (PROC_SPEC)               Size: 2
    * SEQUENTIAL_IO.RESET (PROC_SPEC)               Size: 1
    * SEQUENTIAL_IO.MODE (FUNC_SPEC)                Size: 1
    * SEQUENTIAL_IO.NAME (FUNC_SPEC)                Size: 1
    * SEQUENTIAL_IO.FORM (FUNC_SPEC)                Size: 1
    * SEQUENTIAL_IO.IS_OPEN (FUNC_SPEC)             Size: 1
    * SEQUENTIAL_IO.READ (PROC_SPEC)                Size: 2
    * SEQUENTIAL_IO.WRITE (PROC_SPEC)               Size: 2
    * SEQUENTIAL_IO.END_OF_FILE (FUNC_SPEC)         Size: 1
* DIRECT_IO (GENERIC_PAC)                           Size: 48 Glob_Size: 1 Gen_Size: 1
  Server:
  IO_EXCEPTIONS (PAC_SPEC)                          V 7 G 0 0 0 C 0 :: V 7 G 0 0 0 C 0
  Client:
  SAFE_STREAMS (PAC_BODY)                           V 0 G 1 0 2 C 3 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.FEED_STREAM.SIO (GEN_PAC_INS)        V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
  SAFE_STREAMS.FEED_STREAM (PAC_BODY)               V 0 G 1 0 2 C 3 :: V 0 G 0 0 2 C 0
  SAFE_STREAMS.FEED_STREAM.READ_NEXT (PROC_BODY)    V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.FEED_STREAM.CF (PROC_BODY)           V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  Ctxt:
  IO_EXCEPTIONS
  With:
  IO_EXCEPTIONS
    * DIRECT_IO.CREATE (PROC_SPEC)                  Size: 4
    * DIRECT_IO.OPEN (PROC_SPEC)                    Size: 4
    * DIRECT_IO.CLOSE (PROC_SPEC)                   Size: 1
      Client:
      SAFE_STREAMS (PAC_BODY)                       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
      SAFE_STREAMS.FEED_STREAM (PAC_BODY)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
      SAFE_STREAMS.FEED_STREAM.CF (PROC_BODY)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1

    * DIRECT_IO.DELETE (PROC_SPEC)                  Size: 1
      Client:
      SAFE_STREAMS (PAC_BODY)                       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
      SAFE_STREAMS.FEED_STREAM (PAC_BODY)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
      SAFE_STREAMS.FEED_STREAM.CF (PROC_BODY)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * DIRECT_IO.RESET (PROC_SPEC)                   Size: 2
    * DIRECT_IO.RESET (PROC_SPEC)                   Size: 1
    * DIRECT_IO.MODE (FUNC_SPEC)                    Size: 1
    * DIRECT_IO.NAME (FUNC_SPEC)                    Size: 1
    * DIRECT_IO.FORM (FUNC_SPEC)                    Size: 1
    * DIRECT_IO.IS_OPEN (FUNC_SPEC)                 Size: 1
    * DIRECT_IO.READ (PROC_SPEC)                    Size: 3
      Client:
      SAFE_STREAMS (PAC_BODY)                       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
      SAFE_STREAMS.FEED_STREAM (PAC_BODY)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
      SAFE_STREAMS.FEED_STREAM.READ_NEXT (PROC_BODY) V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * DIRECT_IO.READ (PROC_SPEC)                    Size: 2
    * DIRECT_IO.WRITE (PROC_SPEC)                   Size: 3
    * DIRECT_IO.WRITE (PROC_SPEC)                   Size: 2
    * DIRECT_IO.SET_INDEX (PROC_SPEC)               Size: 2
    * DIRECT_IO.INDEX (FUNC_SPEC)                           Size: 1
    * DIRECT_IO.SIZE (FUNC_SPEC)                            Size: 1
    * DIRECT_IO.END_OF_FILE (FUNC_SPEC)                     Size: 1
* TEXT_IO (PAC_SPEC)                                        Size: 195 Glob_Size: 1
  Server:
  IO_EXCEPTIONS (PAC_SPEC)                                  V 8 G 0 0 0 C 0 :: V 8 G 0 0 0 C 0
  Ctxt:
  IO_EXCEPTIONS
  With:
  IO_EXCEPTIONS
    * TEXT_IO.CREATE (PROC_SPEC)                            Size: 4
    * TEXT_IO.OPEN (PROC_SPEC)                              Size: 4
    * TEXT_IO.CLOSE (PROC_SPEC)                             Size: 1
    * TEXT_IO.DELETE (PROC_SPEC)                            Size: 1
    * TEXT_IO.RESET (PROC_SPEC)                             Size: 2
    * TEXT_IO.RESET (PROC_SPEC)                             Size: 1
    * TEXT_IO.MODE (FUNC_SPEC)                              Size: 1
    * TEXT_IO.NAME (FUNC_SPEC)                              Size: 1
    * TEXT_IO.FORM (FUNC_SPEC)                              Size: 1
    * TEXT_IO.IS_OPEN (FUNC_SPEC)                           Size: 1
    * TEXT_IO.SET_INPUT (PROC_SPEC)                         Size: 1
    * TEXT_IO.SET_OUTPUT (PROC_SPEC)                        Size: 1
    * TEXT_IO.STANDARD_INPUT (FUNC_SPEC)                    Size: 1
    * TEXT_IO.STANDARD_OUTPUT (FUNC_SPEC)                   Size: 1
    * TEXT_IO.CURRENT_INPUT (FUNC_SPEC)                     Size: 1
    * TEXT_IO.CURRENT_OUTPUT (FUNC_SPEC)                    Size: 1
    * TEXT_IO.SET_LINE_LENGTH (PROC_SPEC)                   Size: 2
    * TEXT_IO.SET_LINE_LENGTH (PROC_SPEC)                   Size: 1
    * TEXT_IO.SET_PAGE_LENGTH (PROC_SPEC)                   Size: 2
    * TEXT_IO.SET_PAGE_LENGTH (PROC_SPEC)                   Size: 1
    * TEXT_IO.LINE_LENGTH (FUNC_SPEC)                       Size: 1
    * TEXT_IO.LINE_LENGTH (FUNC_SPEC)                       Size: 1
    * TEXT_IO.PAGE_LENGTH (FUNC_SPEC)                       Size: 1
    * TEXT_IO.PAGE_LENGTH (FUNC_SPEC)                       Size: 1
    * TEXT_IO.NEW_LINE (PROC_SPEC)                          Size: 2
    * TEXT_IO.NEW_LINE (PROC_SPEC)                          Size: 1
    * TEXT_IO.SKIP_LINE (PROC_SPEC)                         Size: 2
    * TEXT_IO.SKIP_LINE (PROC_SPEC)                         Size: 1
    * TEXT_IO.END_OF_LINE (FUNC_SPEC)                       Size: 1
    * TEXT_IO.END_OF_LINE (FUNC_SPEC)                       Size: 1
    * TEXT_IO.NEW_PAGE (PROC_SPEC)                          Size: 1
    * TEXT_IO.NEW_PAGE (PROC_SPEC)                          Size: 1
    * TEXT_IO.SKIP_PAGE (PROC_SPEC)                         Size: 1

    * TEXT_IO.SKIP_PAGE (PROC_SPEC)                         Size: 1
    * TEXT_IO.END_OF_PAGE (FUNC_SPEC)                       Size: 1
    * TEXT_IO.END_OF_PAGE (FUNC_SPEC)                       Size: 1
    * TEXT_IO.END_OF_FILE (FUNC_SPEC)                       Size: 1
    * TEXT_IO.END_OF_FILE (FUNC_SPEC)                       Size: 1
    * TEXT_IO.SET_COL (PROC_SPEC)                           Size: 2
    * TEXT_IO.SET_COL (PROC_SPEC)                           Size: 1
    * TEXT_IO.SET_LINE (PROC_SPEC)                          Size: 2
    * TEXT_IO.SET_LINE (PROC_SPEC)                          Size: 1
    * TEXT_IO.COL (FUNC_SPEC)                               Size: 1
    * TEXT_IO.COL (FUNC_SPEC)                               Size: 1
    * TEXT_IO.LINE (FUNC_SPEC)                              Size: 1
    * TEXT_IO.LINE (FUNC_SPEC)                              Size: 1
    * TEXT_IO.PAGE (FUNC_SPEC)                              Size: 1
    * TEXT_IO.PAGE (FUNC_SPEC)                              Size: 1
    * TEXT_IO.GET (PROC_SPEC)                               Size: 2
    * TEXT_IO.GET (PROC_SPEC)                               Size: 1
    * TEXT_IO.PUT (PROC_SPEC)                               Size: 2
    * TEXT_IO.PUT (PROC_SPEC)                               Size: 1
    * TEXT_IO.GET (PROC_SPEC)                               Size: 2
    * TEXT_IO.GET (PROC_SPEC)                               Size: 1
    * TEXT_IO.PUT (PROC_SPEC)                               Size: 2
    * TEXT_IO.PUT (PROC_SPEC)                               Size: 1
    * TEXT_IO.GET_LINE (PROC_SPEC)                          Size: 3
    * TEXT_IO.GET_LINE (PROC_SPEC)                              Size: 2
    * TEXT_IO.PUT_LINE (PROC_SPEC)                              Size: 2
    * TEXT_IO.PUT_LINE (PROC_SPEC)                              Size: 1
    * TEXT_IO.INTEGER_IO (GENERIC_PAC)                          Size: 22 Gen_Size: 1
      * TEXT_IO.INTEGER_IO.GET (PROC_SPEC)                      Size: 3
      * TEXT_IO.INTEGER_IO.GET (PROC_SPEC)                      Size: 2
      * TEXT_IO.INTEGER_IO.PUT (PROC_SPEC)                      Size: 4
      * TEXT_IO.INTEGER_IO.PUT (PROC_SPEC)                      Size: 3
      * TEXT_IO.INTEGER_IO.GET (PROC_SPEC)                      Size: 3
      * TEXT_IO.INTEGER_IO.PUT (PROC_SPEC)                      Size: 3
    * TEXT_IO.FLOAT_IO (GENERIC_PAC)                            Size: 26 Gen_Size: 1
      * TEXT_IO.FLOAT_IO.GET (PROC_SPEC)                        Size: 3
      * TEXT_IO.FLOAT_IO.GET (PROC_SPEC)                        Size: 2
      * TEXT_IO.FLOAT_IO.PUT (PROC_SPEC)                        Size: 5
      * TEXT_IO.FLOAT_IO.PUT (PROC_SPEC)                        Size: 4
      * TEXT_IO.FLOAT_IO.GET (PROC_SPEC)                        Size: 3
      * TEXT_IO.FLOAT_IO.PUT (PROC_SPEC)                        Size: 4
    * TEXT_IO.FIXED_IO (GENERIC_PAC)                            Size: 26 Gen_Size: 1
      * TEXT_IO.FIXED_IO.GET (PROC_SPEC)                        Size: 3
      * TEXT_IO.FIXED_IO.GET (PROC_SPEC)                        Size: 2
      * TEXT_IO.FIXED_IO.PUT (PROC_SPEC)                        Size: 5
      * TEXT_IO.FIXED_IO.PUT (PROC_SPEC)                        Size: 4
      * TEXT_IO.FIXED_IO.GET (PROC_SPEC)                        Size: 3
      * TEXT_IO.FIXED_IO.PUT (PROC_SPEC)                        Size: 4
    * TEXT_IO.ENUMERATION_IO (GENERIC_PAC)                      Size: 20 Gen_Size: 1
      * TEXT_IO.ENUMERATION_IO.GET (PROC_SPEC)                  Size: 2
      * TEXT_IO.ENUMERATION_IO.GET (PROC_SPEC)                  Size: 1
      * TEXT_IO.ENUMERATION_IO.PUT (PROC_SPEC)                  Size: 4
      * TEXT_IO.ENUMERATION_IO.PUT (PROC_SPEC)                  Size: 3
      * TEXT_IO.ENUMERATION_IO.GET (PROC_SPEC)                  Size: 3
      * TEXT_IO.ENUMERATION_IO.PUT (PROC_SPEC)                  Size: 3
* MACHINE_CODE (PAC_SPEC)                                       Size: 279
    * MACHINE_CODE.W (FUNC_SPEC)                                Size: 1
    * MACHINE_CODE.L (FUNC_SPEC)                                Size: 1
    * MACHINE_CODE.INDR (FUNC_SPEC)                             Size: 1
    * MACHINE_CODE.INCR (FUNC_SPEC)                             Size: 1
    * MACHINE_CODE.DECR (FUNC_SPEC)                             Size: 1
    * MACHINE_CODE.DISP (FUNC_SPEC)                             Size: 2
    * MACHINE_CODE.DISP (FUNC_SPEC)                             Size: 2

    * MACHINE_CODE.INDEX (FUNC_SPEC)                            Size: 5
    * MACHINE_CODE.INDEX (FUNC_SPEC)                            Size: 5
    * MACHINE_CODE.MEMORY (FUNC_SPEC)                           Size: 2
    * MACHINE_CODE.MEMORY (FUNC_SPEC)                           Size: 5
    * MACHINE_CODE.ABSOL (FUNC_SPEC)                            Size: 2
    * MACHINE_CODE.IMMED (FUNC_SPEC)                            Size: 1
    * MACHINE_CODE.IMMED (FUNC_SPEC)                            Size: 1
    * MACHINE_CODE.IMMED (FUNC_SPEC)                            Size: 1
    * MACHINE_CODE."+" (FUNC_SPEC)                              Size: 1
    * MACHINE_CODE."-" (FUNC_SPEC)                              Size: 1
    * MACHINE_CODE."+" (FUNC_SPEC)                              Size: 1
    * MACHINE_CODE."-" (FUNC_SPEC)                              Size: 1
    * MACHINE_CODE."/" (FUNC_SPEC)                              Size: 2
    * MACHINE_CODE.EXT (FUNC_SPEC)                              Size: 2
    * MACHINE_CODE."-" (FUNC_SPEC)                              Size: 2
* UNCHECKED_CONVERSION (GENERIC_FUNC)                           Size: 3 Gen_Size: 2
* UNCHECKED_DEALLOCATION (GENERIC_PROC)                         Size: 3 Gen_Size: 2
* LOW_LEVEL_IO (PAC_SPEC)                                       Size: 8
    * LOW_LEVEL_IO.SEND_CONTROL (PROC_SPEC)                     Size: 2
    * LOW_LEVEL_IO.RECEIVE_CONTROL (PROC_SPEC)                  Size: 2
* CALENDAR (PAC_SPEC)                                           Size: 40
    * CALENDAR.CLOCK (FUNC_SPEC)                                Size: 1
    * CALENDAR.YEAR (FUNC_SPEC)                                 Size: 1
    * CALENDAR.MONTH (FUNC_SPEC)                                Size: 1
    * CALENDAR.DAY (FUNC_SPEC)                                  Size: 1
    * CALENDAR.SECONDS (FUNC_SPEC)                              Size: 1
    * CALENDAR.SPLIT (PROC_SPEC)                                Size: 5
    * CALENDAR.TIME_OF (FUNC_SPEC)                              Size: 4
    * CALENDAR."+" (FUNC_SPEC)                                  Size: 2
    * CALENDAR."+" (FUNC_SPEC)                                  Size: 2
    * CALENDAR."-" (FUNC_SPEC)                                  Size: 2
    * CALENDAR."-" (FUNC_SPEC)                                  Size: 2
    * CALENDAR."<" (FUNC_SPEC)                                  Size: 2
    * CALENDAR."<=" (FUNC_SPEC)                                 Size: 2
    * CALENDAR.">" (FUNC_SPEC)                                  Size: 2
    * CALENDAR.">=" (FUNC_SPEC)                                 Size: 2
* SYSTEM (PAC_SPEC)                                             Size: 49
    * SYSTEM.">" (FUNC_SPEC)                                    Size: 2
    * SYSTEM."<" (FUNC_SPEC)                                    Size: 2
    * SYSTEM.">=" (FUNC_SPEC)                                   Size: 2
    * SYSTEM."<=" (FUNC_SPEC)                                   Size: 2
    * SYSTEM."-" (FUNC_SPEC)                                    Size: 2
    * SYSTEM."+" (FUNC_SPEC)                                    Size: 2
    * SYSTEM."-" (FUNC_SPEC)                                    Size: 2
* GLOBAL_TYPES (PAC_SPEC)                                       Size: 5
  Client:
  LIST_MANAGER (PAC_SPEC)                                       V 2 G 0 0 0 C 0 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                          V 2 G 0 0 0 C 0 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)                    V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_SPEC)                   V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  LIST_MANAGER (PAC_BODY)                                       V 1 G 0 0 0 C 0 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (PAC_BODY)                             V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  SAFE_STREAMS (PAC_SPEC)                                       V 4 G 0 0 0 C 0 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.FEED_STREAM (GENERIC_PAC)                        V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  SAFE_STREAMS (PAC_BODY)                                       V 15 G 0 0 0 C 0 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.FEED_STREAM (PAC_BODY)                           V 2 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_STREAM (PAC_BODY)                          V 7 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  SAFE_STREAMS.GEN_STREAM (GENERIC_PAC)                         V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  SAFE_STREAMS.GEN_STREAM (PAC_BODY)                            V 2 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_2_STREAMS (PAC_BODY)                       V 2 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  SAFE_STREAMS.REDUCE_STREAM (GENERIC_PROC)                     V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  SAFE_STREAMS.OUTPUT_DRIVER (GENERIC_PROC)                     V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  SAFE_STREAMS.OUTPUT_DRIVER (PROC_BODY)                        V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0

  SAFE_STREAMS.REDUCE_STREAM (PROC_BODY)                        V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_2_STREAMS.X_ADVANCE (PROC_BODY)            V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  SAFE_STREAMS.GEN_STREAM.CHECK_BUFFER (PROC_BODY)              V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_STREAM.BUMP (FUNC_BODY)                    V 4 G 0 0 0 C 0 :: V 4 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_STREAM.X_ADVANCE (PROC_BODY)               V 2 G 0 0 0 C 0 :: V 2 G 0 0 0 C 0
  SAFE_STREAMS.FEED_STREAM.CHECK_BUFFER (PROC_BODY)             V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
* LIST_MANAGER (PAC_SPEC)                                       Size: 23 Glob_Size: 2
  Server:
  GLOBAL_TYPES (PAC_SPEC)                                       V 2 G 0 0 0 C 0 :: V 0 G 0 0 0 C 0
  Client:
  SAFE_STREAMS.FEED_STREAM.USER_HANDLE_LIST (GEN_PAC_INS)       V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.FEED_STREAM.STREAM_LIST (GEN_PAC_INS)            V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS (PAC_BODY)                                       V 0 G 14 0 24 C 128 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.FEED_STREAM (PAC_BODY)                           V 0 G 2 0 3 C 21 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MAP_STREAM.MUSER_HANDLE_LIST (GEN_PAC_INS)       V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MAP_STREAM (PAC_BODY)                            V 0 G 1 0 2 C 6 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.FILTER_STREAM.FUSER_HANDLE_LIST (GEN_PAC_INS)    V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.FILTER_STREAM (PAC_BODY)                         V 0 G 1 0 2 C 6 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_STREAM.MUSER_HANDLE_LIST (GEN_PAC_INS)     V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_STREAM.ISTREAM_HANDLE_LIST (GEN_PAC_INS)   V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_STREAM.IUSER_HANDLE_LIST (GEN_PAC_INS)     V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_STREAM (PAC_BODY)                          V 0 G 3 0 4 C 26 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_STREAM.CUSER_HANDLE_LIST (GEN_PAC_INS)    V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_STREAM.ISTREAM_HANDLE_LIST (GEN_PAC_INS)  V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_STREAM (PAC_BODY)                         V 0 G 2 0 3 C 21 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.TRANS_STREAM.TUSER_HANDLE_LIST (GEN_PAC_INS)     V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.TRANS_STREAM (PAC_BODY)                          V 0 G 1 0 2 C 10 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.GEN_STREAM.USER_HANDLE_LIST (GEN_PAC_INS)        V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.GEN_STREAM.STREAM_LIST (GEN_PAC_INS)             V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.GEN_STREAM (PAC_BODY)                            V 0 G 2 0 4 C 20 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_2_STREAMS.MUSER_HANDLE_LIST (GEN_PAC_INS)  V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_2_STREAMS (PAC_BODY)                       V 0 G 1 0 2 C 6 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_2_STREAMS.CUSER_HANDLE_LIST (GEN_PAC_INS) V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_2_STREAMS (PAC_BODY)                      V 0 G 1 0 2 C 12 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_2_STREAMS.X_REGISTER_USER (PROC_BODY)     V 0 G 0 0 1 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MERGE_2_STREAMS.X_REGISTER_USER (PROC_BODY)      V 0 G 0 0 1 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.GEN_STREAM.CHECK_BUFFER (PROC_BODY)              V 0 G 0 0 1 C 8 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.GEN_STREAM.X_REGISTER_USER (PROC_BODY)           V 0 G 0 0 1 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_STREAM.X_REGISTER_USER (PROC_BODY)        V 0 G 0 0 1 C 6 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.TRANS_STREAM.X_REGISTER_USER (PROC_BODY)         V 0 G 0 0 1 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MERGE_STREAM.X_REGISTER_USER (PROC_BODY)         V 0 G 0 0 1 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.FILTER_STREAM.X_REGISTER_USER (PROC_BODY)        V 0 G 0 0 1 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MAP_STREAM.X_REGISTER_USER (PROC_BODY)           V 0 G 0 0 1 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.FEED_STREAM.X_REGISTER_USER (PROC_BODY)          V 0 G 0 0 1 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_2_STREAMS.X_CLOSE_STREAM (PROC_BODY)      V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MERGE_2_STREAMS.X_CLOSE_STREAM (PROC_BODY)       V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.GEN_STREAM.X_CLOSE_STREAM (PROC_BODY)            V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_STREAM.SET_UP_NEXT_STREAM (PROC_BODY)     V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_STREAM.X_CLOSE_STREAM (PROC_BODY)         V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MERGE_STREAM.DRU (PROC_BODY)                     V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MERGE_STREAM.X_CLOSE_STREAM (PROC_BODY)          V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.FILTER_STREAM.X_CLOSE_STREAM (PROC_BODY)         V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MAP_STREAM.X_CLOSE_STREAM (PROC_BODY)            V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.FEED_STREAM.CHECK_BUFFER (PROC_BODY)             V 0 G 0 0 0 C 8 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.FEED_STREAM.X_CLOSE_STREAM (PROC_BODY)           V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.GEN_STREAM.X_ADVANCE (PROC_BODY)                 V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_STREAM.X_ADVANCE (PROC_BODY)              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MERGE_STREAM.X_ADVANCE (PROC_BODY)               V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.FEED_STREAM.X_ADVANCE (PROC_BODY)                V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_2_STREAMS.SET_UP_NEXT_STREAM (PROC_BODY)  V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_2_STREAMS.OK_TO_SET_UP (FUNC_BODY)        V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_STREAM.OK_TO_SET_UP (FUNC_BODY)           V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.TRANS_STREAM.CHECK_USERS (FUNC_BODY)             V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0

SAFE_STREAMS.TRANS_STREAM.RESET_USERS (PROC_BODY)             V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MERGE_STREAM.RU (PROC_BODY)                      V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MERGE_STREAM.X_MERGE (PROC_BODY)                 V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.GEN_STREAM.READ_NEXT (PROC_BODY)                 V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_STREAM.X_CONCAT (PROC_BODY)               V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.FEED_STREAM.READ_NEXT (PROC_BODY)                V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.FEED_STREAM.X_FEED (PROC_BODY)                   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
Ctxt:
GLOBAL_TYPES
With:
GLOBAL_TYPES
  * LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                      Size: 19 Gen_Size: 1
    Server:
    GLOBAL_TYPES (PAC_SPEC)                                   V 2 G 0 0 0 C 0 :: V 0 G 0 0 0 C 0
    Client:
    SAFE_STREAMS.FEED_STREAM.USER_HANDLE_LIST (GEN_PAC_INS)   V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
    SAFE_STREAMS.FEED_STREAM.STREAM_LIST (GEN_PAC_INS)        V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
    SAFE_STREAMS (PAC_BODY)                                   V 0 G 14 0 24 C 128 :: V 0 G 0 0 0 C 0
    SAFE_STREAMS.FEED_STREAM (PAC_BODY)                       V 0 G 2 0 3 C 21 :: V 0 G 0 0 2 C 0
    SAFE_STREAMS.MAP_STREAM.MUSER_HANDLE_LIST (GEN_PAC_INS)   V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
    SAFE_STREAMS.MAP_STREAM (PAC_BODY)                        V 0 G 1 0 2 C 6 :: V 0 G 0 0 1 C 0
    SAFE_STREAMS.FILTER_STREAM.FUSER_HANDLE_LIST (GEN_PAC_INS)V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
    SAFE_STREAMS.FILTER_STREAM (PAC_BODY)                     V 0 G 1 0 2 C 6 :: V 0 G 0 0 1 C 0
    SAFE_STREAMS.MERGE_STREAM.MUSER_HANDLE_LIST (GEN_PAC_INS) V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
    SAFE_STREAMS.MERGE_STREAM.ISTREAM_HANDLE_LIST (GEN_PAC_INS)
                                                              V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
    SAFE_STREAMS.MERGE_STREAM.IUSER_HANDLE_LIST (GEN_PAC_INS) V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
    SAFE_STREAMS.MERGE_STREAM (PAC_BODY)                      V 0 G 3 0 4 C 26 :: V 0 G 0 0 3 C 0
    SAFE_STREAMS.CONCAT_STREAM.CUSER_HANDLE_LIST (GEN_PAC_INS)V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
    SAFE_STREAMS.CONCAT_STREAM.ISTREAM_HANDLE_LIST (GEN_PAC_INS)
                                                              V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
    SAFE_STREAMS.CONCAT_STREAM (PAC_BODY)                     V 0 G 2 0 3 C 21 :: V 0 G 0 0 2 C 0
    SAFE_STREAMS.TRANS_STREAM.TUSER_HANDLE_LIST (GEN_PAC_INS) V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
    SAFE_STREAMS.TRANS_STREAM (PAC_BODY)                      V 0 G 1 0 2 C 10 :: V 0 G 0 0 1 C 0
    SAFE_STREAMS.GEN_STREAM.USER_HANDLE_LIST (GEN_PAC_INS)    V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
    SAFE_STREAMS.GEN_STREAM.STREAM_LIST (GEN_PAC_INS)         V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
SAFE_STREAMS.GEN_STREAM (PAC_BODY)                     V 0 G 2 0 4 C 20 :: V 0 G 0 0 2 C 0
SAFE_STREAMS.MERGE_2_STREAMS.MUSER_HANDLE_LIST (GEN_PAC_INS)
                                                       V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
SAFE_STREAMS.MERGE_2_STREAMS (PAC_BODY)                V 0 G 1 0 2 C 6 :: V 0 G 0 0 1 C 0
SAFE_STREAMS.CONCAT_2_STREAMS.CUSER_HANDLE_LIST (GEN_PAC_INS)
                                                       V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
SAFE_STREAMS.CONCAT_2_STREAMS (PAC_BODY)               V 0 G 1 0 2 C 12 :: V 0 G 0 0 1 C 0
SAFE_STREAMS.CONCAT_2_STREAMS.X_REGISTER_USER (PROC_BODY) V 0 G 0 0 1 C 4 :: V 0 G 0 0 1 C 0
SAFE_STREAMS.MERGE_2_STREAMS.X_REGISTER_USER (PROC_BODY)  V 0 G 0 0 1 C 4 :: V 0 G 0 0 1 C 0
SAFE_STREAMS.GEN_STREAM.CHECK_BUFFER (PROC_BODY)       V 0 G 0 0 1 C 8 :: V 0 G 0 0 1 C 0
SAFE_STREAMS.GEN_STREAM.X_REGISTER_USER (PROC_BODY)    V 0 G 0 0 1 C 4 :: V 0 G 0 0 1 C 0
SAFE_STREAMS.CONCAT_STREAM.X_REGISTER_USER (PROC_BODY) V 0 G 0 0 1 C 6 :: V 0 G 0 0 1 C 0
SAFE_STREAMS.TRANS_STREAM.X_REGISTER_USER (PROC_BODY)  V 0 G 0 0 1 C 4 :: V 0 G 0 0 1 C 0
SAFE_STREAMS.MERGE_STREAM.X_REGISTER_USER (PROC_BODY)  V 0 G 0 0 1 C 4 :: V 0 G 0 0 1 C 0
SAFE_STREAMS.FILTER_STREAM.X_REGISTER_USER (PROC_BODY) V 0 G 0 0 1 C 4 :: V 0 G 0 0 1 C 0
SAFE_STREAMS.MAP_STREAM.X_REGISTER_USER (PROC_BODY)    V 0 G 0 0 1 C 4 :: V 0 G 0 0 1 C 0
SAFE_STREAMS.FEED_STREAM.X_REGISTER_USER (PROC_BODY)   V 0 G 0 0 1 C 4 :: V 0 G 0 0 1 C 0
SAFE_STREAMS.CONCAT_2_STREAMS.X_CLOSE_STREAM (PROC_BODY)  V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MERGE_2_STREAMS.X_CLOSE_STREAM (PROC_BODY)   V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.GEN_STREAM.X_CLOSE_STREAM (PROC_BODY)     V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_STREAM.SET_UP_NEXT_STREAM (PROC_BODY) V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_STREAM.X_CLOSE_STREAM (PROC_BODY)  V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MERGE_STREAM.DRU (PROC_BODY)              V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MERGE_STREAM.X_CLOSE_STREAM (PROC_BODY)   V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.FILTER_STREAM.X_CLOSE_STREAM (PROC_BODY)  V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MAP_STREAM.X_CLOSE_STREAM (PROC_BODY)     V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.FEED_STREAM.CHECK_BUFFER (PROC_BODY)      V 0 G 0 0 0 C 8 :: V 0 G 0 0 0 C 0

SAFE_STREAMS.FEED_STREAM.X_CLOSE_STREAM (PROC_BODY)    V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.GEN_STREAM.X_ADVANCE (PROC_BODY)          V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_STREAM.X_ADVANCE (PROC_BODY)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MERGE_STREAM.X_ADVANCE (PROC_BODY)        V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.FEED_STREAM.X_ADVANCE (PROC_BODY)         V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_2_STREAMS.SET_UP_NEXT_STREAM (PROC_BODY)
                                                       V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_2_STREAMS.OK_TO_SET_UP (FUNC_BODY) V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_STREAM.OK_TO_SET_UP (FUNC_BODY)    V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.TRANS_STREAM.CHECK_USERS (FUNC_BODY)      V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.TRANS_STREAM.RESET_USERS (PROC_BODY)      V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MERGE_STREAM.RU (PROC_BODY)               V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.MERGE_STREAM.X_MERGE (PROC_BODY)          V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.GEN_STREAM.READ_NEXT (PROC_BODY)          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.CONCAT_STREAM.X_CONCAT (PROC_BODY)        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.FEED_STREAM.READ_NEXT (PROC_BODY)         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
SAFE_STREAMS.FEED_STREAM.X_FEED (PROC_BODY)            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
* LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)          Size: 2
  Client:
  LIST_MANAGER.LIST_TYPE.UNION_ELEM (FUNC_BODY)        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.UNION_LIST (FUNC_BODY)        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS (PAC_BODY)                              V 0 G 0 0 0 C 25 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_2_STREAMS (PAC_BODY)             V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_2_STREAMS.X_REGISTER_USER (PROC_BODY)
                                                       V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  SAFE_STREAMS.MERGE_2_STREAMS (PAC_BODY)              V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_2_STREAMS.X_REGISTER_USER (PROC_BODY)V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  SAFE_STREAMS.GEN_STREAM (PAC_BODY)                   V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.GEN_STREAM.READ_NEXT (PROC_BODY)        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.GEN_STREAM.X_REGISTER_USER (PROC_BODY)  V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  SAFE_STREAMS.CONCAT_STREAM (PAC_BODY)                V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_STREAM.X_REGISTER_USER (PROC_BODY)  V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  SAFE_STREAMS.CONCAT_STREAM.X_CONCAT (PROC_BODY)      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.TRANS_STREAM (PAC_BODY)                 V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.TRANS_STREAM.X_REGISTER_USER (PROC_BODY)   V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  SAFE_STREAMS.MERGE_STREAM (PAC_BODY)                 V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_STREAM.RU (PROC_BODY)             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_STREAM.X_REGISTER_USER (PROC_BODY)   V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  SAFE_STREAMS.MERGE_STREAM.X_MERGE (PROC_BODY)        V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  SAFE_STREAMS.FILTER_STREAM (PAC_BODY)                V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.FILTER_STREAM.X_REGISTER_USER (PROC_BODY)  V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  SAFE_STREAMS.MAP_STREAM (PAC_BODY)                      V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MAP_STREAM.X_REGISTER_USER (PROC_BODY)     V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  SAFE_STREAMS.FEED_STREAM (PAC_BODY)                     V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.FEED_STREAM.READ_NEXT (PROC_BODY)          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.FEED_STREAM.X_FEED (PROC_BODY)             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.FEED_STREAM.X_REGISTER_USER (PROC_BODY)    V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
* LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)              Size: 1
  Server:
  GLOBAL_TYPES (PAC_SPEC)                                 V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  Client:
  LIST_MANAGER.LIST_TYPE.IS_MEMBER (FUNC_BODY)            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.UNION_LIST (FUNC_BODY)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS (PAC_BODY)                                 V 0 G 0 0 0 C 37 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_2_STREAMS (PAC_BODY)                V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_2_STREAMS.SET_UP_NEXT_STREAM (PROC_BODY)
                                                          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.CONCAT_2_STREAMS.OK_TO_SET_UP (FUNC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.CONCAT_2_STREAMS.X_REGISTER_USER (PROC_BODY)
                                                          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.CONCAT_2_STREAMS.X_CLOSE_STREAM (PROC_BODY)V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_2_STREAMS (PAC_BODY)                 V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_2_STREAMS.X_REGISTER_USER (PROC_BODY)V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1

  SAFE_STREAMS.MERGE_2_STREAMS.X_CLOSE_STREAM (PROC_BODY) V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.GEN_STREAM (PAC_BODY)                      V 0 G 0 0 0 C 6 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.GEN_STREAM.CHECK_BUFFER (PROC_BODY)        V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 3
  SAFE_STREAMS.GEN_STREAM.X_REGISTER_USER (PROC_BODY)     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.GEN_STREAM.X_CLOSE_STREAM (PROC_BODY)      V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  SAFE_STREAMS.CONCAT_STREAM (PAC_BODY)                   V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_STREAM.SET_UP_NEXT_STREAM (PROC_BODY)
                                                          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.CONCAT_STREAM.OK_TO_SET_UP (FUNC_BODY)     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.CONCAT_STREAM.X_REGISTER_USER (PROC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.CONCAT_STREAM.X_CLOSE_STREAM (PROC_BODY)   V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  SAFE_STREAMS.TRANS_STREAM (PAC_BODY)                    V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.TRANS_STREAM.CHECK_USERS (FUNC_BODY)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.TRANS_STREAM.RESET_USERS (PROC_BODY)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.TRANS_STREAM.X_REGISTER_USER (PROC_BODY)   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_STREAM (PAC_BODY)                    V 0 G 0 0 0 C 7 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_STREAM.RU (PROC_BODY)                V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_STREAM.DRU (PROC_BODY)               V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_STREAM.X_ADVANCE (PROC_BODY)         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_STREAM.X_REGISTER_USER (PROC_BODY)   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_STREAM.X_CLOSE_STREAM (PROC_BODY)    V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  SAFE_STREAMS.MERGE_STREAM.X_MERGE (PROC_BODY)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.FILTER_STREAM (PAC_BODY)                   V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.FILTER_STREAM.X_REGISTER_USER (PROC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.FILTER_STREAM.X_CLOSE_STREAM (PROC_BODY)   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MAP_STREAM (PAC_BODY)                      V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MAP_STREAM.X_REGISTER_USER (PROC_BODY)     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MAP_STREAM.X_CLOSE_STREAM (PROC_BODY)      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.FEED_STREAM (PAC_BODY)                     V 0 G 0 0 0 C 6 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.FEED_STREAM.CHECK_BUFFER (PROC_BODY)       V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 3
  SAFE_STREAMS.FEED_STREAM.X_REGISTER_USER (PROC_BODY)    V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.FEED_STREAM.X_CLOSE_STREAM (PROC_BODY)     V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
* LIST_MANAGER.LIST_TYPE.IS_MEMBER (FUNC_SPEC)            Size: 2
  Client:
  LIST_MANAGER.LIST_TYPE.UNION_ELEM (FUNC_BODY)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* LIST_MANAGER.LIST_TYPE.UNION_ELEM (FUNC_SPEC)           Size: 2
* LIST_MANAGER.LIST_TYPE.UNION_LIST (FUNC_SPEC)           Size: 2
* LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)           Size: 1
  Client:
  LIST_MANAGER.LIST_TYPE.IS_MEMBER (FUNC_BODY)            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.UNION_LIST (FUNC_BODY)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_BODY)         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS (PAC_BODY)                                 V 0 G 0 0 0 C 13 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_2_STREAMS (PAC_BODY)                V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_2_STREAMS.SET_UP_NEXT_STREAM (PROC_BODY)
                                                     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.CONCAT_2_STREAMS.OK_TO_SET_UP (FUNC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.GEN_STREAM (PAC_BODY)                 V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.GEN_STREAM.CHECK_BUFFER (PROC_BODY)   V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  SAFE_STREAMS.CONCAT_STREAM (PAC_BODY)              V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_STREAM.SET_UP_NEXT_STREAM (PROC_BODY)
                                                     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.CONCAT_STREAM.OK_TO_SET_UP (FUNC_BODY)     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.TRANS_STREAM (PAC_BODY)               V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.TRANS_STREAM.CHECK_USERS (FUNC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.TRANS_STREAM.RESET_USERS (PROC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_STREAM (PAC_BODY)               V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_STREAM.RU (PROC_BODY)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_STREAM.DRU (PROC_BODY)          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_STREAM.X_MERGE (PROC_BODY)      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.FEED_STREAM (PAC_BODY)                V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.FEED_STREAM.CHECK_BUFFER (PROC_BODY)  V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2

* LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)   Size: 2
  Client:
  LIST_MANAGER.LIST_TYPE.IS_MEMBER (FUNC_BODY)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.UNION_LIST (FUNC_BODY)      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS (PAC_BODY)                            V 0 G 0 0 0 C 13 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_2_STREAMS (PAC_BODY)           V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_2_STREAMS.SET_UP_NEXT_STREAM (PROC_BODY)
                                                     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.CONCAT_2_STREAMS.OK_TO_SET_UP (FUNC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.GEN_STREAM (PAC_BODY)                 V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.GEN_STREAM.CHECK_BUFFER (PROC_BODY)   V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  SAFE_STREAMS.CONCAT_STREAM (PAC_BODY)              V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_STREAM.SET_UP_NEXT_STREAM (PROC_BODY)
                                                     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.CONCAT_STREAM.OK_TO_SET_UP (FUNC_BODY)     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.TRANS_STREAM (PAC_BODY)               V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.TRANS_STREAM.CHECK_USERS (FUNC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.TRANS_STREAM.RESET_USERS (PROC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_STREAM (PAC_BODY)               V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_STREAM.RU (PROC_BODY)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_STREAM.DRU (PROC_BODY)          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_STREAM.X_MERGE (PROC_BODY)      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.FEED_STREAM (PAC_BODY)                V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.FEED_STREAM.CHECK_BUFFER (PROC_BODY)  V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
* LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_SPEC)        Size: 2
  Server:
  GLOBAL_TYPES (PAC_SPEC)                            V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  Client:
  SAFE_STREAMS (PAC_BODY)                            V 0 G 0 0 0 C 15 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.GEN_STREAM (PAC_BODY)                 V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.GEN_STREAM.X_ADVANCE (PROC_BODY)      V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 3
  SAFE_STREAMS.CONCAT_STREAM (PAC_BODY)              V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_STREAM.SET_UP_NEXT_STREAM (PROC_BODY)
                                                     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.CONCAT_STREAM.X_ADVANCE (PROC_BODY)   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.CONCAT_STREAM.X_REGISTER_USER (PROC_BODY)  V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  SAFE_STREAMS.CONCAT_STREAM.X_CLOSE_STREAM (PROC_BODY)   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_STREAM (PAC_BODY)               V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.MERGE_STREAM.X_ADVANCE (PROC_BODY)    V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 4
  SAFE_STREAMS.FEED_STREAM (PAC_BODY)                V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.FEED_STREAM.X_ADVANCE (PROC_BODY)     V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 3
* LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)    Size: 2
  Client:
  SAFE_STREAMS (PAC_BODY)                            V 0 G 0 0 0 C 25 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_2_STREAMS (PAC_BODY)           V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.CONCAT_2_STREAMS.X_REGISTER_USER (PROC_BODY)
                                                     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.CONCAT_2_STREAMS.X_CLOSE_STREAM (PROC_BODY)V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_2_STREAMS (PAC_BODY)            V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
```

```
        SAFE_STREAMS.MERGE_2_STREAMS.X_REGISTER_USER (PROC_BODY)V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        SAFE_STREAMS.MERGE_2_STREAMS.X_CLOSE_STREAM (PROC_BODY) V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        SAFE_STREAMS.GEN_STREAM (PAC_BODY)                      V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
        SAFE_STREAMS.GEN_STREAM.CHECK_BUFFER (PROC_BODY)        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        SAFE_STREAMS.GEN_STREAM.X_REGISTER_USER (PROC_BODY)     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        SAFE_STREAMS.GEN_STREAM.X_CLOSE_STREAM (PROC_BODY)      V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
        SAFE_STREAMS.CONCAT_STREAM (PAC_BODY)                   V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
        SAFE_STREAMS.CONCAT_STREAM.SET_UP_NEXT_STREAM (PROC_BODY)
                                                                V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        SAFE_STREAMS.CONCAT_STREAM.X_REGISTER_USER (PROC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        SAFE_STREAMS.CONCAT_STREAM.X_CLOSE_STREAM (PROC_BODY)   V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
        SAFE_STREAMS.TRANS_STREAM (PAC_BODY)                    V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
        SAFE_STREAMS.TRANS_STREAM.X_REGISTER_USER (PROC_BODY)   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        SAFE_STREAMS.MERGE_STREAM (PAC_BODY)                    V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0

        SAFE_STREAMS.MERGE_STREAM.DRU (PROC_BODY)               V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        SAFE_STREAMS.MERGE_STREAM.X_REGISTER_USER (PROC_BODY)   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        SAFE_STREAMS.MERGE_STREAM.X_CLOSE_STREAM (PROC_BODY)    V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
        SAFE_STREAMS.FILTER_STREAM (PAC_BODY)                   V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
        SAFE_STREAMS.FILTER_STREAM.X_REGISTER_USER (PROC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        SAFE_STREAMS.FILTER_STREAM.X_CLOSE_STREAM (PROC_BODY)   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        SAFE_STREAMS.MAP_STREAM (PAC_BODY)                      V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
        SAFE_STREAMS.MAP_STREAM.X_REGISTER_USER (PROC_BODY)     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        SAFE_STREAMS.MAP_STREAM.X_CLOSE_STREAM (PROC_BODY)      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        SAFE_STREAMS.FEED_STREAM (PAC_BODY)                     V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
        SAFE_STREAMS.FEED_STREAM.CHECK_BUFFER (PROC_BODY)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        SAFE_STREAMS.FEED_STREAM.X_REGISTER_USER (PROC_BODY)    V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        SAFE_STREAMS.FEED_STREAM.X_CLOSE_STREAM (PROC_BODY)     V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  * LIST_MANAGER (PAC_BODY)                                     Size: 90
    Server:
    GLOBAL_TYPES (PAC_SPEC)                                     V 1 G 0 0 0 C 0 :: V 0 G 0 0 0 C 0
    * LIST_MANAGER.LIST_TYPE (PAC_BODY)                         Size: 89
      Server:
      GLOBAL_TYPES (PAC_SPEC)                                   V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
      * LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_BODY)             Size: 14
      * LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_BODY)              Size: 2
      * LIST_MANAGER.LIST_TYPE.IS_MEMBER (FUNC_BODY)            Size: 12
        Server:
        LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
      * LIST_MANAGER.LIST_TYPE.UNION_ELEM (FUNC_BODY)           Size: 8
        Server:
        LIST_MANAGER.LIST_TYPE.IS_MEMBER (FUNC_SPEC)            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
      * LIST_MANAGER.LIST_TYPE.UNION_LIST (FUNC_BODY)           Size: 12
        Server:
        LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
        LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
      * LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_BODY)           Size: 2
      * LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_BODY)        Size: 6
      * LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_BODY)             Size: 11
      * LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_BODY)         Size: 11
        Server:
        LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS (PAC_SPEC)                                       Size: 290 Glob_Size: 1
  Server:
  GLOBAL_TYPES (PAC_SPEC)                                       V 4 G 0 0 0 C 0 :: V 0 G 0 0 0 C 0
  Ctxt:
  GLOBAL_TYPES
  With:
  GLOBAL_TYPES
    * SAFE_STREAMS.FEED_STREAM (GENERIC_PAC)                    Size: 23 Gen_Size: 2
      Server:
      GLOBAL_TYPES (PAC_SPEC)                                   V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
      * SAFE_STREAMS.FEED_STREAM.ADVANCE (PROC_SPEC)            Size: 4
  * SAFE_STREAMS.FEED_STREAM.FEED (PROC_SPEC)            Size: 3
  * SAFE_STREAMS.FEED_STREAM.REGISTER_USER (PROC_SPEC)   Size: 3
  * SAFE_STREAMS.FEED_STREAM.OPEN_STREAM (PROC_SPEC)     Size: 4
  * SAFE_STREAMS.FEED_STREAM.OPEN_STREAM (PROC_SPEC)     Size: 2
  * SAFE_STREAMS.FEED_STREAM.CLOSE_STREAM (PROC_SPEC)    Size: 1
  * SAFE_STREAMS.FEED_STREAM.SET_EOS (PROC_SPEC)         Size: 1

* SAFE_STREAMS.MAP_STREAM (GENERIC_PAC)                  Size: 25 Gen_Size: 12
  * SAFE_STREAMS.MAP_STREAM.F (FUNC_SPEC)                Size: 1
    Client:
    SAFE_STREAMS.MAP_STREAM.X_ADVANCE (PROC_BODY)        V O G O O O C 1 :: V O G O O O C 1
  * SAFE_STREAMS.MAP_STREAM.IA (PROC_SPEC)               Size: 4
    Client:
    SAFE_STREAMS.MAP_STREAM.X_ADVANCE (PROC_BODY)        V O G O O O C 1 :: V O G O O O C 1
  * SAFE_STREAMS.MAP_STREAM.IRU (PROC_SPEC)              Size: 3
    Client:
    SAFE_STREAMS.MAP_STREAM.X_REGISTER_USER (PROC_BODY)  V O G O O O C 2 :: V O G O O O C 2
    SAFE_STREAMS.MAP_STREAM.X_CLOSE_STREAM (PROC_BODY)   V O G O O O C 1 :: V O G O O O C 1
  * SAFE_STREAMS.MAP_STREAM.OPEN_STREAM (PROC_SPEC)      Size: 2
  * SAFE_STREAMS.MAP_STREAM.ADVANCE (PROC_SPEC)          Size: 4
  * SAFE_STREAMS.MAP_STREAM.REGISTER_USER (PROC_SPEC)    Size: 3
  * SAFE_STREAMS.MAP_STREAM.CLOSE_STREAM (PROC_SPEC)     Size: 1
* SAFE_STREAMS.FILTER_STREAM (GENERIC_PAC)               Size: 24 Gen_Size: 11
  * SAFE_STREAMS.FILTER_STREAM.F (FUNC_SPEC)             Size: 1
    Client:
    SAFE_STREAMS.FILTER_STREAM.X_ADVANCE (PROC_BODY)     V O G O O O C 1 :: V O G O O O C 1
  * SAFE_STREAMS.FILTER_STREAM.IA (PROC_SPEC)            Size: 4
    Client:
    SAFE_STREAMS.FILTER_STREAM.X_ADVANCE (PROC_BODY)     V O G O O O C 1 :: V O G O O O C 1
  * SAFE_STREAMS.FILTER_STREAM.IRU (PROC_SPEC)           Size: 3
    Client:
    SAFE_STREAMS.FILTER_STREAM.X_REGISTER_USER (PROC_BODY)  V O G O O O C 2 :: V O G O O O C 2
    SAFE_STREAMS.FILTER_STREAM.X_CLOSE_STREAM (PROC_BODY)  V O G O O O C 1 :: V O G O O O C 1
  * SAFE_STREAMS.FILTER_STREAM.OPEN_STREAM (PROC_SPEC)   Size: 2
  * SAFE_STREAMS.FILTER_STREAM.ADVANCE (PROC_SPEC)       Size: 4
  * SAFE_STREAMS.FILTER_STREAM.REGISTER_USER (PROC_SPEC)  Size: 3
  * SAFE_STREAMS.FILTER_STREAM.CLOSE_STREAM (PROC_SPEC)  Size: 1
* SAFE_STREAMS.MERGE_STREAM (GENERIC_PAC)                Size: 23 Gen_Size: 10
  * SAFE_STREAMS.MERGE_STREAM.IA (PROC_SPEC)             Size: 4
    Client:
    SAFE_STREAMS.MERGE_STREAM.X_ADVANCE (PROC_BODY)      V O G O O O C 1 :: V O G O O O C 1
  * SAFE_STREAMS.MERGE_STREAM.IRU (PROC_SPEC)            Size: 3
    Client:
    SAFE_STREAMS.MERGE_STREAM.RU (PROC_BODY)             V O G O O O C 1 :: V O G O O O C 1
    SAFE_STREAMS.MERGE_STREAM.DRU (PROC_BODY)            V O G O O O C 1 :: V O G O O O C 1
    SAFE_STREAMS.MERGE_STREAM.X_MERGE (PROC_BODY)        V O G O O O C 1 :: V O G O O O C 1
  * SAFE_STREAMS.MERGE_STREAM.MERGE (PROC_SPEC)          Size: 2
  * SAFE_STREAMS.MERGE_STREAM.ADVANCE (PROC_SPEC)        Size: 4
  * SAFE_STREAMS.MERGE_STREAM.REGISTER_USER (PROC_SPEC)  Size: 3
  * SAFE_STREAMS.MERGE_STREAM.CLOSE_STREAM (PROC_SPEC)   Size: 1
* SAFE_STREAMS.CONCAT_STREAM (GENERIC_PAC)               Size: 23 Gen_Size: 10
  * SAFE_STREAMS.CONCAT_STREAM.IA (PROC_SPEC)            Size: 4
    Client:
    SAFE_STREAMS.CONCAT_STREAM.X_ADVANCE (PROC_BODY)     V O G O O O C 1 :: V O G O O O C 1
  * SAFE_STREAMS.CONCAT_STREAM.IRU (PROC_SPEC)           Size: 3
    Client:
    SAFE_STREAMS.CONCAT_STREAM.SET_UP_NEXT_STREAM (PROC_BODY)
                                                         V O G O O O C 2 :: V O G O O O C 2
    SAFE_STREAMS.CONCAT_STREAM.X_REGISTER_USER (PROC_BODY)  V O G O O O C 2 :: V O G O O O C 2
    SAFE_STREAMS.CONCAT_STREAM.X_CLOSE_STREAM (PROC_BODY)  V O G O O O C 1 :: V O G O O O C 1
  * SAFE_STREAMS.CONCAT_STREAM.CONCAT (PROC_SPEC)        Size: 2
  * SAFE_STREAMS.CONCAT_STREAM.ADVANCE (PROC_SPEC)       Size: 4
  * SAFE_STREAMS.CONCAT_STREAM.REGISTER_USER (PROC_SPEC)  Size: 3
  * SAFE_STREAMS.CONCAT_STREAM.CLOSE_STREAM (PROC_SPEC)  Size: 1
* SAFE_STREAMS.TRANS_STREAM (GENERIC_PAC)                Size: 27 Gen_Size: 13
  * SAFE_STREAMS.TRANS_STREAM.UPDATE (PROC_SPEC)         Size: 2
    Client:
    SAFE_STREAMS.TRANS_STREAM.X_ADVANCE (PROC_BODY)      V O G O O O C 1 :: V O G O O O C 1
  * SAFE_STREAMS.TRANS_STREAM.IA (PROC_SPEC)              Size: 4
    Client:
    SAFE_STREAMS.TRANS_STREAM.X_ADVANCE (PROC_BODY)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.TRANS_STREAM.IRU (PROC_SPEC)             Size: 3
    Client:
    SAFE_STREAMS.TRANS_STREAM.X_OPEN_STREAM (PROC_BODY)   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    SAFE_STREAMS.TRANS_STREAM.X_CLOSE_STREAM (PROC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.TRANS_STREAM.OPEN_STREAM (PROC_SPEC)     Size: 3
  * SAFE_STREAMS.TRANS_STREAM.ADVANCE (PROC_SPEC)         Size: 4
  * SAFE_STREAMS.TRANS_STREAM.REGISTER_USER (PROC_SPEC)   Size: 3
  * SAFE_STREAMS.TRANS_STREAM.CLOSE_STREAM (PROC_SPEC)    Size: 1
* SAFE_STREAMS.GEN_STREAM (GENERIC_PAC)                   Size: 17 Gen_Size: 4
  Server:
  GLOBAL_TYPES (PAC_SPEC)                                 V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  * SAFE_STREAMS.GEN_STREAM.MY_ADVANCE (PROC_SPEC)        Size: 2
    Client:
    SAFE_STREAMS.GEN_STREAM.READ_NEXT (PROC_BODY)         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.GEN_STREAM.OPEN_STREAM (PROC_SPEC)       Size: 2
  * SAFE_STREAMS.GEN_STREAM.ADVANCE (PROC_SPEC)           Size: 4
  * SAFE_STREAMS.GEN_STREAM.REGISTER_USER (PROC_SPEC)     Size: 3
  * SAFE_STREAMS.GEN_STREAM.CLOSE_STREAM (PROC_SPEC)      Size: 1
* SAFE_STREAMS.ENUM_STREAM (GENERIC_PAC)                  Size: 26 Gen_Size: 1
  * SAFE_STREAMS.ENUM_STREAM.OPEN_STREAM (PROC_SPEC)      Size: 3
  * SAFE_STREAMS.ENUM_STREAM.ADVANCE (PROC_SPEC)          Size: 4
  * SAFE_STREAMS.ENUM_STREAM.REGISTER_USER (PROC_SPEC)    Size: 3
  * SAFE_STREAMS.ENUM_STREAM.CLOSE_STREAM (PROC_SPEC)     Size: 1
* SAFE_STREAMS.MERGE_2_STREAMS (GENERIC_PAC)              Size: 33 Gen_Size: 19
  * SAFE_STREAMS.MERGE_2_STREAMS.IA1 (PROC_SPEC)          Size: 4
    Client:
    SAFE_STREAMS.MERGE_2_STREAMS.X_ADVANCE (PROC_BODY)    V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_2_STREAMS.IRU1 (PROC_SPEC)         Size: 3
    Client:
    SAFE_STREAMS.MERGE_2_STREAMS.RU (PROC_BODY)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_2_STREAMS.IA2 (PROC_SPEC)          Size: 4
    Client:
    SAFE_STREAMS.MERGE_2_STREAMS.X_ADVANCE (PROC_BODY)    V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_2_STREAMS.IRU2 (PROC_SPEC)         Size: 3
    Client:
    SAFE_STREAMS.MERGE_2_STREAMS.RU (PROC_BODY)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_2_STREAMS.OPEN_STREAM (PROC_SPEC)   Size: 3
  * SAFE_STREAMS.MERGE_2_STREAMS.ADVANCE (PROC_SPEC)       Size: 4
  * SAFE_STREAMS.MERGE_2_STREAMS.REGISTER_USER (PROC_SPEC) Size: 3
  * SAFE_STREAMS.MERGE_2_STREAMS.CLOSE_STREAM (PROC_SPEC)  Size: 1
* SAFE_STREAMS.CONCAT_2_STREAMS (GENERIC_PAC)             Size: 33 Gen_Size: 19
  * SAFE_STREAMS.CONCAT_2_STREAMS.IA1 (PROC_SPEC)         Size: 4
    Client:
    SAFE_STREAMS.CONCAT_2_STREAMS.X_ADVANCE (PROC_BODY)   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_2_STREAMS.IRU1 (PROC_SPEC)        Size: 3
    Client:
    SAFE_STREAMS.CONCAT_2_STREAMS.SET_UP_NEXT_STREAM (PROC_BODY)
                                                          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    SAFE_STREAMS.CONCAT_2_STREAMS.X_REGISTER_USER (PROC_BODY)
                                                          V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
    SAFE_STREAMS.CONCAT_2_STREAMS.X_CLOSE_STREAM (PROC_BODY)V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_2_STREAMS.IA2 (PROC_SPEC)         Size: 4
    Client:
    SAFE_STREAMS.CONCAT_2_STREAMS.X_ADVANCE (PROC_BODY)   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_2_STREAMS.IRU2 (PROC_SPEC)        Size: 3
    Client:
    SAFE_STREAMS.CONCAT_2_STREAMS.SET_UP_NEXT_STREAM (PROC_BODY)
                                                          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    SAFE_STREAMS.CONCAT_2_STREAMS.X_REGISTER_USER (PROC_BODY)
                                                          V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2

    SAFE_STREAMS.CONCAT_2_STREAMS.X_CLOSE_STREAM (PROC_BODY)V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_2_STREAMS.OPEN_STREAM (PROC_SPEC)   Size: 3
  * SAFE_STREAMS.CONCAT_2_STREAMS.ADVANCE (PROC_SPEC)       Size: 4
  * SAFE_STREAMS.CONCAT_2_STREAMS.REGISTER_USER (PROC_SPEC) Size: 3
```

```
    * SAFE_STREAMS.CONCAT_2_STREAMS.CLOSE_STREAM (PROC_SPEC)  Size: 1
  * SAFE_STREAMS.REDUCE_STREAM (GENERIC_PROC)               Size: 16 Gen_Size: 13
    Server:
    GLOBAL_TYPES (PAC_SPEC)                                 V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
    * SAFE_STREAMS.REDUCE_STREAM.UPDATE (PROC_SPEC)         Size: 2
    * SAFE_STREAMS.REDUCE_STREAM.IA (PROC_SPEC)             Size: 4
    * SAFE_STREAMS.REDUCE_STREAM.IRU (PROC_SPEC)            Size: 3
  * SAFE_STREAMS.OUTPUT_DRIVER (GENERIC_PROC)               Size: 16 Gen_Size: 13
    Server:
    GLOBAL_TYPES (PAC_SPEC)                                 V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
    * SAFE_STREAMS.OUTPUT_DRIVER.WRITE (PROC_SPEC)          Size: 2
    * SAFE_STREAMS.OUTPUT_DRIVER.IA (PROC_SPEC)             Size: 4
    * SAFE_STREAMS.OUTPUT_DRIVER.IRU (PROC_SPEC)            Size: 3
* SAFE_STREAMS (PAC_BODY)                                   Size: 1789 Glob_Size: 3
  Server:
  GLOBAL_TYPES (PAC_SPEC)                                   V 15 G 0 0 0 C 0 :: V 0 G 0 0 0 C 0
  LIST_MANAGER (PAC_SPEC)                                   V 0 G 14 0 24 C 128 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                      V 0 G 14 0 24 C 128 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)                V 0 G 0 0 0 C 37 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)           V 0 G 0 0 0 C 25 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)               V 0 G 0 0 0 C 25 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)             V 0 G 0 0 0 C 13 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)          V 0 G 0 0 0 C 13 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_SPEC)               V 0 G 0 0 0 C 15 :: V 0 G 0 0 0 C 0
  DIRECT_IO (GENERIC_PAC)                                   V 0 G 1 0 2 C 3 :: V 0 G 0 0 0 C 0
  DIRECT_IO.READ (PROC_SPEC)                                V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
  DIRECT_IO.CLOSE (PROC_SPEC)                               V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
  DIRECT_IO.DELETE (PROC_SPEC)                              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
  Ctxt:
  GLOBAL_TYPES
  DIRECT_IO
  LIST_MANAGER
  With:
  LIST_MANAGER
  DIRECT_IO
  * SAFE_STREAMS.FEED_STREAM (PAC_BODY)                     Size: 268
    Server:
    GLOBAL_TYPES (PAC_SPEC)                                 V 2 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
    DIRECT_IO (GENERIC_PAC)                                 V 0 G 1 0 2 C 3 :: V 0 G 0 0 2 C 0
    LIST_MANAGER (PAC_SPEC)                                 V 0 G 2 0 3 C 21 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 2 0 3 C 21 :: V 0 G 0 0 2 C 0
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)              V 0 G 0 0 0 C 6 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)         V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)             V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_SPEC)             V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)           V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)        V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
    DIRECT_IO.READ (PROC_SPEC)                              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
    DIRECT_IO.CLOSE (PROC_SPEC)                             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
    DIRECT_IO.DELETE (PROC_SPEC)                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
    * SAFE_STREAMS.FEED_STREAM.USER_HANDLE_LIST (GEN_PAC_INS) Size: 1
      Gen_Ins     : LIST_MANAGER.LIST_TYPE
      Server:
      LIST_MANAGER (PAC_SPEC)                               V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
      LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                  V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
    * SAFE_STREAMS.FEED_STREAM.STREAM_LIST (GEN_PAC_INS)    Size: 1
      Gen_Ins     : LIST_MANAGER.LIST_TYPE
      Server:
      LIST_MANAGER (PAC_SPEC)                               V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0

      LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                  V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
    * SAFE_STREAMS.FEED_STREAM.SIO (GEN_PAC_INS)            Size: 1
      Gen_Ins     : DIRECT_IO
      Server:
      DIRECT_IO (GENERIC_PAC)                               V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
    * SAFE_STREAMS.FEED_STREAM.STREAM_GUARD (TASK_TYP)      Size: 19
      * SAFE_STREAMS.FEED_STREAM.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                            Size: 4
    Client:
    SAFE_STREAMS.FEED_STREAM.ADVANCE (PROC_BODY)          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FEED_STREAM.STREAM_GUARD.FEED (ENTRY_SPEC)
                                                          Size: 3
    Client:
    SAFE_STREAMS.FEED_STREAM.FEED (PROC_BODY)             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FEED_STREAM.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                          Size: 3
    Client:
    SAFE_STREAMS.FEED_STREAM.REGISTER_USER (PROC_BODY)    V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FEED_STREAM.STREAM_GUARD.OPEN_STREAM (ENTRY_SPEC)
                                                          Size: 2
    Client:
    SAFE_STREAMS.FEED_STREAM.OPEN_STREAM (PROC_BODY)      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FEED_STREAM.STREAM_GUARD.FOPEN_STREAM (ENTRY_SPEC)
                                                          Size: 4
    Client:
    SAFE_STREAMS.FEED_STREAM.OPEN_STREAM (PROC_BODY)      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FEED_STREAM.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                          Size: 1
    Client:
    SAFE_STREAMS.FEED_STREAM.CLOSE_STREAM (PROC_BODY)     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FEED_STREAM.STREAM_GUARD.SET_EOS (ENTRY_SPEC)
                                                          Size: 1
    Client:
    SAFE_STREAMS.FEED_STREAM.SET_EOS (PROC_BODY)          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS.FEED_STREAM.CF (PROC_BODY)                 Size: 7
  Server:
  DIRECT_IO (GENERIC_PAC)                                 V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  DIRECT_IO.CLOSE (PROC_SPEC)                             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  DIRECT_IO.DELETE (PROC_SPEC)                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS.FEED_STREAM.FD (FUNC_BODY)                 Size: 7
* SAFE_STREAMS.FEED_STREAM.READ_NEXT (PROC_BODY)          Size: 15
  Server:
  DIRECT_IO (GENERIC_PAC)                                 V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
  DIRECT_IO.READ (PROC_SPEC)                              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER (PAC_SPEC)                                 V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS.FEED_STREAM.OK_TO_ADVANCE (FUNC_BODY)      Size: 9
* SAFE_STREAMS.FEED_STREAM.CHECK_BUFFER (PROC_BODY)       Size: 17
  Server:
  LIST_MANAGER (PAC_SPEC)                                 V 0 G 0 0 0 C 8 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 0 0 0 C 8 :: V 0 G 0 0 0 C 0
  GLOBAL_TYPES (PAC_SPEC)                                 V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)              V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 3
  LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)           V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)        V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS.FEED_STREAM.X_ADVANCE (PROC_BODY)          Size: 36
  Server:
  LIST_MANAGER (PAC_SPEC)                                 V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_SPEC)             V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 3

* SAFE_STREAMS.FEED_STREAM.X_FEED (PROC_BODY)             Size: 14
  Server:
  LIST_MANAGER (PAC_SPEC)                                 V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS.FEED_STREAM.X_REGISTER_USER (PROC_BODY)    Size: 22
  Server:
  LIST_MANAGER (PAC_SPEC)                                 V 0 G 0 0 1 C 4 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 0 0 1 C 4 :: V 0 G 0 0 1 C 0
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)             V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS.FEED_STREAM.X_OPEN_STREAM (PROC_BODY)      Size: 6
```

```
  * SAFE_STREAMS.FEED_STREAM.X_FOPEN_STREAM (PROC_BODY)     Size: 13
  * SAFE_STREAMS.FEED_STREAM.X_SET_EOS (PROC_BODY)          Size: 6
  * SAFE_STREAMS.FEED_STREAM.X_CLOSE_STREAM (PROC_BODY)     Size: 14
    Server:
    LIST_MANAGER (PAC_SPEC)                                 V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)              V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
    LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)         V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  * SAFE_STREAMS.FEED_STREAM.ADVANCE (PROC_BODY)            Size: 7
    Server:
    SAFE_STREAMS.FEED_STREAM.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FEED_STREAM.FEED (PROC_BODY)               Size: 6
    Server:
    SAFE_STREAMS.FEED_STREAM.STREAM_GUARD.FEED (ENTRY_SPEC) V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FEED_STREAM.REGISTER_USER (PROC_BODY)      Size: 6
    Server:
    SAFE_STREAMS.FEED_STREAM.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FEED_STREAM.OPEN_STREAM (PROC_BODY)        Size: 5
    Server:
    SAFE_STREAMS.FEED_STREAM.STREAM_GUARD.OPEN_STREAM (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FEED_STREAM.OPEN_STREAM (PROC_BODY)        Size: 7
    Server:
    SAFE_STREAMS.FEED_STREAM.STREAM_GUARD.FOPEN_STREAM (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FEED_STREAM.CLOSE_STREAM (PROC_BODY)       Size: 4
    Server:
    SAFE_STREAMS.FEED_STREAM.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FEED_STREAM.SET_EOS (PROC_BODY)            Size: 4
    Server:
    SAFE_STREAMS.FEED_STREAM.STREAM_GUARD.SET_EOS (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FEED_STREAM.STREAM_GUARD (TASK_BODY)       Size: 20
* SAFE_STREAMS.MAP_STREAM (PAC_BODY)                        Size: 120
  Server:
  LIST_MANAGER (PAC_SPEC)                                   V 0 G 1 0 2 C 6 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                      V 0 G 1 0 2 C 6 :: V 0 G 0 0 1 C 0
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)                V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)           V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)               V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  * SAFE_STREAMS.MAP_STREAM.MUSER_HANDLE_LIST (GEN_PAC_INS) Size: 1
    Gen_Ins     : LIST_MANAGER.LIST_TYPE
    Server:
    LIST_MANAGER (PAC_SPEC)                                 V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
  * SAFE_STREAMS.MAP_STREAM.STREAM_GUARD (TASK_TYP)         Size: 11

    * SAFE_STREAMS.MAP_STREAM.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                            Size: 4
      Client:
      SAFE_STREAMS.MAP_STREAM.ADVANCE (PROC_BODY)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.MAP_STREAM.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                            Size: 3
      Client:
      SAFE_STREAMS.MAP_STREAM.REGISTER_USER (PROC_BODY)     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.MAP_STREAM.STREAM_GUARD.OPEN_STREAM (ENTRY_SPEC)
                                                            Size: 2
      Client:
      SAFE_STREAMS.MAP_STREAM.OPEN_STREAM (PROC_BODY)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.MAP_STREAM.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                            Size: 1
      Client:
      SAFE_STREAMS.MAP_STREAM.CLOSE_STREAM (PROC_BODY)      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MAP_STREAM.X_ADVANCE (PROC_BODY)           Size: 20
    Server:
```

```
    SAFE_STREAMS.MAP_STREAM.IA (PROC_SPEC)               V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    SAFE_STREAMS.MAP_STREAM.F (FUNC_SPEC)                V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MAP_STREAM.X_REGISTER_USER (PROC_BODY)     Size: 24
    Server:
    LIST_MANAGER (PAC_SPEC)                              V 0 G 0 0 1 C 4 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                 V 0 G 0 0 1 C 4 :: V 0 G 0 0 1 C 0
    SAFE_STREAMS.MAP_STREAM.IRU (PROC_SPEC)              V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
    LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)          V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MAP_STREAM.X_OPEN_STREAM (PROC_BODY)       Size: 6
  * SAFE_STREAMS.MAP_STREAM.X_CLOSE_STREAM (PROC_BODY)      Size: 11
    Server:
    LIST_MANAGER (PAC_SPEC)                              V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                 V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    SAFE_STREAMS.MAP_STREAM.IRU (PROC_SPEC)              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MAP_STREAM.ADVANCE (PROC_BODY)             Size: 7
    Server:
    SAFE_STREAMS.MAP_STREAM.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MAP_STREAM.REGISTER_USER (PROC_BODY)       Size: 6
    Server:
    SAFE_STREAMS.MAP_STREAM.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MAP_STREAM.OPEN_STREAM (PROC_BODY)         Size: 5
    Server:
    SAFE_STREAMS.MAP_STREAM.STREAM_GUARD.OPEN_STREAM (ENTRY_SPEC)
                                                         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MAP_STREAM.CLOSE_STREAM (PROC_BODY)        Size: 4
    Server:
    SAFE_STREAMS.MAP_STREAM.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MAP_STREAM.STREAM_GUARD (TASK_BODY)        Size: 14
* SAFE_STREAMS.FILTER_STREAM (PAC_BODY)                     Size: 124
  Server:
  LIST_MANAGER (PAC_SPEC)                                V 0 G 1 0 2 C 6 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                   V 0 G 1 0 2 C 6 :: V 0 G 0 0 1 C 0
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)             V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)        V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)            V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0

  * SAFE_STREAMS.FILTER_STREAM.FUSER_HANDLE_LIST (GEN_PAC_INS)
                                                         Size: 1
    Gen_Ins     : LIST_MANAGER.LIST_TYPE
    Server:
    LIST_MANAGER (PAC_SPEC)                              V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                 V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
  * SAFE_STREAMS.FILTER_STREAM.STREAM_GUARD (TASK_TYP)      Size: 11
    * SAFE_STREAMS.FILTER_STREAM.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                         Size: 4
      Client:
      SAFE_STREAMS.FILTER_STREAM.ADVANCE (PROC_BODY)     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.FILTER_STREAM.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                         Size: 3
      Client:
      SAFE_STREAMS.FILTER_STREAM.REGISTER_USER (PROC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.FILTER_STREAM.STREAM_GUARD.OPEN_STREAM (ENTRY_SPEC)
                                                         Size: 2
      Client:
      SAFE_STREAMS.FILTER_STREAM.OPEN_STREAM (PROC_BODY)    V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.FILTER_STREAM.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                         Size: 1
      Client:
      SAFE_STREAMS.FILTER_STREAM.CLOSE_STREAM (PROC_BODY)   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FILTER_STREAM.X_ADVANCE (PROC_BODY)        Size: 24
    Server:
```

```
    SAFE_STREAMS.FILTER_STREAM.IA (PROC_SPEC)             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    SAFE_STREAMS.FILTER_STREAM.F (FUNC_SPEC)              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FILTER_STREAM.X_REGISTER_USER (PROC_BODY)  Size: 24
    Server:
    LIST_MANAGER (PAC_SPEC)                               V 0 G 0 0 1 C 4 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                  V 0 G 0 0 1 C 4 :: V 0 G 0 0 1 C 0
    SAFE_STREAMS.FILTER_STREAM.IRU (PROC_SPEC)            V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
    LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)           V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FILTER_STREAM.X_OPEN_STREAM (PROC_BODY)    Size: 6
  * SAFE_STREAMS.FILTER_STREAM.X_CLOSE_STREAM (PROC_BODY)   Size: 11
    Server:
    LIST_MANAGER (PAC_SPEC)                               V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                  V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    SAFE_STREAMS.FILTER_STREAM.IRU (PROC_SPEC)            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FILTER_STREAM.ADVANCE (PROC_BODY)          Size: 7
    Server:
    SAFE_STREAMS.FILTER_STREAM.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FILTER_STREAM.REGISTER_USER (PROC_BODY)    Size: 6
    Server:
    SAFE_STREAMS.FILTER_STREAM.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FILTER_STREAM.OPEN_STREAM (PROC_BODY)      Size: 5
    Server:
    SAFE_STREAMS.FILTER_STREAM.STREAM_GUARD.OPEN_STREAM (ENTRY_SPEC)
                                                          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FILTER_STREAM.CLOSE_STREAM (PROC_BODY)     Size: 4
    Server:
    SAFE_STREAMS.FILTER_STREAM.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.FILTER_STREAM.STREAM_GUARD (TASK_BODY)     Size: 14
* SAFE_STREAMS.MERGE_STREAM (PAC_BODY)                      Size: 181
  Server:
  LIST_MANAGER (PAC_SPEC)                                 V 0 G 3 0 4 C 26 :: V 0 G 0 0 0 C 0

  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 3 0 4 C 26 :: V 0 G 0 0 3 C 0
  GLOBAL_TYPES (PAC_SPEC)                                 V 7 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)             V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)           V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)              V 0 G 0 0 0 C 7 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)        V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)         V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_SPEC)             V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
  * SAFE_STREAMS.MERGE_STREAM.MUSER_HANDLE_LIST (GEN_PAC_INS)
                                                          Size: 1
    Gen_Ins     : LIST_MANAGER.LIST_TYPE
    Server:
    LIST_MANAGER (PAC_SPEC)                               V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                  V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
  * SAFE_STREAMS.MERGE_STREAM.ISTREAM_HANDLE_LIST (GEN_PAC_INS)
                                                          Size: 1
    Gen_Ins     : LIST_MANAGER.LIST_TYPE
    Server:
    LIST_MANAGER (PAC_SPEC)                               V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                  V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
  * SAFE_STREAMS.MERGE_STREAM.IUSER_HANDLE_LIST (GEN_PAC_INS)
                                                          Size: 1
    Gen_Ins     : LIST_MANAGER.LIST_TYPE
    Server:
    LIST_MANAGER (PAC_SPEC)                               V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                  V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
  * SAFE_STREAMS.MERGE_STREAM.STREAM_GUARD (TASK_TYP)       Size: 11
    * SAFE_STREAMS.MERGE_STREAM.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                          Size: 4
    Client:
    SAFE_STREAMS.MERGE_STREAM.ADVANCE (PROC_BODY)        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_STREAM.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                         Size: 3
    Client:
    SAFE_STREAMS.MERGE_STREAM.REGISTER_USER (PROC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_STREAM.STREAM_GUARD.MERGE (ENTRY_SPEC)
                                                         Size: 2
    Client:
    SAFE_STREAMS.MERGE_STREAM.MERGE (PROC_BODY)          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_STREAM.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                         Size: 1
    Client:
    SAFE_STREAMS.MERGE_STREAM.CLOSE_STREAM (PROC_BODY)   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS.MERGE_STREAM.BUMP (FUNC_BODY)             Size: 7
  Server:
  GLOBAL_TYPES (PAC_SPEC)                                V 4 G 0 0 0 C 0 :: V 4 G 0 0 0 C 0
* SAFE_STREAMS.MERGE_STREAM.RU (PROC_BODY)               Size: 10
  Server:
  LIST_MANAGER (PAC_SPEC)                                V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                   V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_STREAM.IRU (PROC_SPEC)              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS.MERGE_STREAM.DRU (PROC_BODY)              Size: 10
  Server:
  LIST_MANAGER (PAC_SPEC)                                V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                   V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1

  SAFE_STREAMS.MERGE_STREAM.IRU (PROC_SPEC)              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS.MERGE_STREAM.X_ADVANCE (PROC_BODY)        Size: 36
  Server:
  LIST_MANAGER (PAC_SPEC)                                V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                   V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
  GLOBAL_TYPES (PAC_SPEC)                                V 2 G 0 0 0 C 0 :: V 2 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_SPEC)            V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 4
  SAFE_STREAMS.MERGE_STREAM.IA (PROC_SPEC)               V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS.MERGE_STREAM.X_REGISTER_USER (PROC_BODY)  Size: 24
  Server:
  LIST_MANAGER (PAC_SPEC)                                V 0 G 0 0 1 C 4 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                   V 0 G 0 0 1 C 4 :: V 0 G 0 0 1 C 0
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)            V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS.MERGE_STREAM.X_CLOSE_STREAM (PROC_BODY)   Size: 14
  Server:
  LIST_MANAGER (PAC_SPEC)                                V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                   V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)             V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)        V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
* SAFE_STREAMS.MERGE_STREAM.X_MERGE (PROC_BODY)          Size: 18
  Server:
  LIST_MANAGER (PAC_SPEC)                                V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                   V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)            V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  SAFE_STREAMS.MERGE_STREAM.IRU (PROC_SPEC)              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS.MERGE_STREAM.ADVANCE (PROC_BODY)          Size: 7
  Server:
    SAFE_STREAMS.MERGE_STREAM.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_STREAM.REGISTER_USER (PROC_BODY)     Size: 6
    Server:
    SAFE_STREAMS.MERGE_STREAM.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_STREAM.MERGE (PROC_BODY)             Size: 5
    Server:
    SAFE_STREAMS.MERGE_STREAM.STREAM_GUARD.MERGE (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_STREAM.CLOSE_STREAM (PROC_BODY)      Size: 4
    Server:
    SAFE_STREAMS.MERGE_STREAM.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_STREAM.STREAM_GUARD (TASK_BODY)      Size: 14
* SAFE_STREAMS.CONCAT_STREAM (PAC_BODY)                     Size: 168
  Server:
  LIST_MANAGER (PAC_SPEC)                                   V 0 G 2 0 3 C 21 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                      V 0 G 2 0 3 C 21 :: V 0 G 0 0 2 C 0
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)               V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_SPEC)               V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)                V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)           V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)             V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)          V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  * SAFE_STREAMS.CONCAT_STREAM.CUSER_HANDLE_LIST (GEN_PAC_INS)
                                                            Size: 1
    Gen_Ins     : LIST_MANAGER.LIST_TYPE
    Server:
    LIST_MANAGER (PAC_SPEC)                                 V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0

    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
  * SAFE_STREAMS.CONCAT_STREAM.ISTREAM_HANDLE_LIST (GEN_PAC_INS)
                                                            Size: 1
    Gen_Ins     : LIST_MANAGER.LIST_TYPE
    Server:
    LIST_MANAGER (PAC_SPEC)                                 V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
  * SAFE_STREAMS.CONCAT_STREAM.STREAM_GUARD (TASK_TYP)      Size: 11
    * SAFE_STREAMS.CONCAT_STREAM.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                            Size: 4
      Client:
      SAFE_STREAMS.CONCAT_STREAM.ADVANCE (PROC_BODY)        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.CONCAT_STREAM.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                            Size: 3
      Client:
      SAFE_STREAMS.CONCAT_STREAM.REGISTER_USER (PROC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.CONCAT_STREAM.STREAM_GUARD.CONCAT (ENTRY_SPEC)
                                                            Size: 2
      Client:
      SAFE_STREAMS.CONCAT_STREAM.CONCAT (PROC_BODY)         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.CONCAT_STREAM.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                            Size: 1
      Client:
      SAFE_STREAMS.CONCAT_STREAM.CLOSE_STREAM (PROC_BODY)   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_STREAM.SET_UP_NEXT_STREAM (PROC_BODY)
                                                            Size: 14
    Server:
    LIST_MANAGER (PAC_SPEC)                                 V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 0 0 0 C 5 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_SPEC)             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    SAFE_STREAMS.CONCAT_STREAM.IRU (PROC_SPEC)              V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  * SAFE_STREAMS.CONCAT_STREAM.OK_TO_SET_UP (FUNC_BODY)     Size: 10
    Server:
```

```
    LIST_MANAGER (PAC_SPEC)                                V O G 0 0 0 C 3 :: V O G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                   V O G 0 0 0 C 3 :: V O G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)          V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)             V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)       V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_STREAM.X_ADVANCE (PROC_BODY)       Size: 32
    Server:
    LIST_MANAGER (PAC_SPEC)                                V O G 0 0 0 C 1 :: V O G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                   V O G 0 0 0 C 1 :: V O G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_SPEC)            V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
    SAFE_STREAMS.CONCAT_STREAM.IA (PROC_SPEC)              V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_STREAM.X_REGISTER_USER (PROC_BODY)  Size: 27
    Server:
    LIST_MANAGER (PAC_SPEC)                                V O G 0 0 1 C 6 :: V O G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                   V O G 0 0 1 C 6 :: V O G 0 0 1 C 0
    LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)            V O G 0 0 0 C 2 :: V O G 0 0 0 C 2
    LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_SPEC)            V O G 0 0 0 C 2 :: V O G 0 0 0 C 2
    SAFE_STREAMS.CONCAT_STREAM.IRU (PROC_SPEC)             V O G 0 0 0 C 2 :: V O G 0 0 0 C 2
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)             V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)        V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_STREAM.X_CLOSE_STREAM (PROC_BODY)   Size: 15
    Server:
    LIST_MANAGER (PAC_SPEC)                                V O G 0 0 0 C 5 :: V O G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                   V O G 0 0 0 C 5 :: V O G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_SPEC)            V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)             V O G 0 0 0 C 2 :: V O G 0 0 0 C 2

    LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)        V O G 0 0 0 C 2 :: V O G 0 0 0 C 2
    SAFE_STREAMS.CONCAT_STREAM.IRU (PROC_SPEC)             V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_STREAM.X_CONCAT (PROC_BODY)        Size: 9
    Server:
    LIST_MANAGER (PAC_SPEC)                                V O G 0 0 0 C 1 :: V O G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                   V O G 0 0 0 C 1 :: V O G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)            V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_STREAM.ADVANCE (PROC_BODY)         Size: 7
    Server:
    SAFE_STREAMS.CONCAT_STREAM.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                           V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_STREAM.REGISTER_USER (PROC_BODY)   Size: 6
    Server:
    SAFE_STREAMS.CONCAT_STREAM.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                           V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_STREAM.CONCAT (PROC_BODY)          Size: 5
    Server:
    SAFE_STREAMS.CONCAT_STREAM.STREAM_GUARD.CONCAT (ENTRY_SPEC)
                                                           V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_STREAM.CLOSE_STREAM (PROC_BODY)    Size: 4
    Server:
    SAFE_STREAMS.CONCAT_STREAM.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                           V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_STREAM.STREAM_GUARD (TASK_BODY)    Size: 14
* SAFE_STREAMS.TRANS_STREAM (PAC_BODY)                     Size: 146
  Server:
  LIST_MANAGER (PAC_SPEC)                                  V O G 1 0 2 C 10 :: V O G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                     V O G 1 0 2 C 10 :: V O G 0 0 1 C 0
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)              V O G 0 0 0 C 2 :: V O G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)               V O G 0 0 0 C 3 :: V O G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)          V O G 0 0 0 C 1 :: V O G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)            V O G 0 0 0 C 2 :: V O G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)         V O G 0 0 0 C 2 :: V O G 0 0 0 C 0
  * SAFE_STREAMS.TRANS_STREAM.TUSER_HANDLE_LIST (GEN_PAC_INS)
                                                           Size: 1
    Gen_Ins     : LIST_MANAGER.LIST_TYPE
    Server:
    LIST_MANAGER (PAC_SPEC)                                V O G 1 0 0 C 0 :: V O G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                   V O G 1 0 0 C 0 :: V O G 1 0 0 C 0
  * SAFE_STREAMS.TRANS_STREAM.STREAM_GUARD (TASK_TYP)      Size: 12
    * SAFE_STREAMS.TRANS_STREAM.STREAM_GUARD.OPEN_STREAM (ENTRY_SPEC)
```

```
                                                     Size: 3
      Client:
      SAFE_STREAMS.TRANS_STREAM.OPEN_STREAM (PROC_BODY)     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.TRANS_STREAM.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                     Size: 4
      Client:
      SAFE_STREAMS.TRANS_STREAM.ADVANCE (PROC_BODY)         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.TRANS_STREAM.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                     Size: 3
      Client:
      SAFE_STREAMS.TRANS_STREAM.REGISTER_USER (PROC_BODY)   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.TRANS_STREAM.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                     Size: 1
      Client:
      SAFE_STREAMS.TRANS_STREAM.CLOSE_STREAM (PROC_BODY)    V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.TRANS_STREAM.CHECK_USERS (FUNC_BODY)       Size: 8
    Server:
    LIST_MANAGER (PAC_SPEC)                                 V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1

  * SAFE_STREAMS.TRANS_STREAM.RESET_USERS (PROC_BODY)       Size: 7
    Server:
    LIST_MANAGER (PAC_SPEC)                                 V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.TRANS_STREAM.X_ADVANCE (PROC_BODY)         Size: 28
    Server:
    SAFE_STREAMS.TRANS_STREAM.IA (PROC_SPEC)                V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    SAFE_STREAMS.TRANS_STREAM.UPDATE (PROC_SPEC)            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.TRANS_STREAM.X_REGISTER_USER (PROC_BODY)   Size: 22
    Server:
    LIST_MANAGER (PAC_SPEC)                                 V 0 G 0 0 1 C 4 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 0 0 1 C 4 :: V 0 G 0 0 1 C 0
    LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)             V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.TRANS_STREAM.X_OPEN_STREAM (PROC_BODY)     Size: 9
    Server:
    SAFE_STREAMS.TRANS_STREAM.IRU (PROC_SPEC)               V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.TRANS_STREAM.X_CLOSE_STREAM (PROC_BODY)    Size: 8
    Server:
    SAFE_STREAMS.TRANS_STREAM.IRU (PROC_SPEC)               V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.TRANS_STREAM.ADVANCE (PROC_BODY)           Size: 7
    Server:
    SAFE_STREAMS.TRANS_STREAM.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.TRANS_STREAM.REGISTER_USER (PROC_BODY)     Size: 6
    Server:
    SAFE_STREAMS.TRANS_STREAM.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.TRANS_STREAM.OPEN_STREAM (PROC_BODY)       Size: 6
    Server:
    SAFE_STREAMS.TRANS_STREAM.STREAM_GUARD.OPEN_STREAM (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.TRANS_STREAM.CLOSE_STREAM (PROC_BODY)      Size: 4
    Server:
    SAFE_STREAMS.TRANS_STREAM.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.TRANS_STREAM.STREAM_GUARD (TASK_BODY)      Size: 14
* SAFE_STREAMS.GEN_STREAM (PAC_BODY)                        Size: 177
  Server:
  LIST_MANAGER (PAC_SPEC)                                   V 0 G 2 0 4 C 20 :: V 0 G 0 0 0 C 0
  GLOBAL_TYPES (PAC_SPEC)                                   V 2 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
```

```
LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                      V 0 G 2 0 4 C 20 :: V 0 G 0 0 2 C 0
LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)                V 0 G 0 0 0 C 6 :: V 0 G 0 0 0 C 0
LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)           V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)               V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_SPEC)               V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)             V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)          V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
* SAFE_STREAMS.GEN_STREAM.USER_HANDLE_LIST (GEN_PAC_INS)  Size: 1
  Gen_Ins     : LIST_MANAGER.LIST_TYPE
  Server:
  LIST_MANAGER (PAC_SPEC)                                 V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
* SAFE_STREAMS.GEN_STREAM.STREAM_LIST (GEN_PAC_INS)       Size: 1
  Gen_Ins     : LIST_MANAGER.LIST_TYPE
  Server:
  LIST_MANAGER (PAC_SPEC)                                 V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
* SAFE_STREAMS.GEN_STREAM.STREAM_GUARD (TASK_TYP)         Size: 11

  * SAFE_STREAMS.GEN_STREAM.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                          Size: 4
    Client:
    SAFE_STREAMS.GEN_STREAM.ADVANCE (PROC_BODY)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.GEN_STREAM.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                          Size: 3
    Client:
    SAFE_STREAMS.GEN_STREAM.REGISTER_USER (PROC_BODY)     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.GEN_STREAM.STREAM_GUARD.OPEN_STREAM (ENTRY_SPEC)
                                                          Size: 2
    Client:
    SAFE_STREAMS.GEN_STREAM.OPEN_STREAM (PROC_BODY)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.GEN_STREAM.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                          Size: 1
    Client:
    SAFE_STREAMS.GEN_STREAM.CLOSE_STREAM (PROC_BODY)      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS.GEN_STREAM.READ_NEXT (PROC_BODY)           Size: 10
  Server:
  LIST_MANAGER (PAC_SPEC)                                 V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 0
  SAFE_STREAMS.GEN_STREAM.MY_ADVANCE (PROC_SPEC)          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)             V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS.GEN_STREAM.OK_TO_ADVANCE (FUNC_BODY)       Size: 9
* SAFE_STREAMS.GEN_STREAM.CHECK_BUFFER (PROC_BODY)        Size: 18
  Server:
  LIST_MANAGER (PAC_SPEC)                                 V 0 G 0 0 1 C 8 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 0 0 1 C 8 :: V 0 G 0 0 1 C 0
  GLOBAL_TYPES (PAC_SPEC)                                 V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)              V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 3
  LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)           V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)        V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS.GEN_STREAM.X_ADVANCE (PROC_BODY)           Size: 36
  Server:
  LIST_MANAGER (PAC_SPEC)                                 V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.NTH_ELEM (FUNC_SPEC)             V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 3
* SAFE_STREAMS.GEN_STREAM.X_REGISTER_USER (PROC_BODY)     Size: 22
  Server:
  LIST_MANAGER (PAC_SPEC)                                 V 0 G 0 0 1 C 4 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 0 0 1 C 4 :: V 0 G 0 0 1 C 0
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)             V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
* SAFE_STREAMS.GEN_STREAM.X_OPEN_STREAM (PROC_BODY)       Size: 6
* SAFE_STREAMS.GEN_STREAM.X_CLOSE_STREAM (PROC_BODY)      Size: 14
  Server:
  LIST_MANAGER (PAC_SPEC)                                 V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
```

```
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)              V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
    LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)         V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
  * SAFE_STREAMS.GEN_STREAM.ADVANCE (PROC_BODY)             Size: 7
    Server:
    SAFE_STREAMS.GEN_STREAM.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.GEN_STREAM.REGISTER_USER (PROC_BODY)       Size: 6
    Server:
    SAFE_STREAMS.GEN_STREAM.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.GEN_STREAM.OPEN_STREAM (PROC_BODY)         Size: 5
    Server:
    SAFE_STREAMS.GEN_STREAM.STREAM_GUARD.OPEN_STREAM (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1

  * SAFE_STREAMS.GEN_STREAM.CLOSE_STREAM (PROC_BODY)        Size: 4
    Server:
    SAFE_STREAMS.GEN_STREAM.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.GEN_STREAM.STREAM_GUARD (TASK_BODY)        Size: 14
* SAFE_STREAMS.MERGE_2_STREAMS (PAC_BODY)                   Size: 157
  Server:
  LIST_MANAGER (PAC_SPEC)                                   V 0 G 1 0 2 C 6 :: V 0 G 0 0 0 C 0
  GLOBAL_TYPES (PAC_SPEC)                                   V 2 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                      V 0 G 1 0 2 C 6 :: V 0 G 0 0 1 C 0
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)                V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)           V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)               V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  * SAFE_STREAMS.MERGE_2_STREAMS.MUSER_HANDLE_LIST (GEN_PAC_INS)
                                                            Size: 1
    Gen_Ins     : LIST_MANAGER.LIST_TYPE
    Server:
    LIST_MANAGER (PAC_SPEC)                                 V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
  * SAFE_STREAMS.MERGE_2_STREAMS.STREAM_GUARD (TASK_TYP)    Size: 12
    * SAFE_STREAMS.MERGE_2_STREAMS.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                            Size: 4
      Client:
      SAFE_STREAMS.MERGE_2_STREAMS.ADVANCE (PROC_BODY)      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.MERGE_2_STREAMS.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                            Size: 3
      Client:
      SAFE_STREAMS.MERGE_2_STREAMS.REGISTER_USER (PROC_BODY)V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.MERGE_2_STREAMS.STREAM_GUARD.OPEN_STREAM (ENTRY_SPEC)
                                                            Size: 3
      Client:
      SAFE_STREAMS.MERGE_2_STREAMS.OPEN_STREAM (PROC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.MERGE_2_STREAMS.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                            Size: 1
      Client:
      SAFE_STREAMS.MERGE_2_STREAMS.CLOSE_STREAM (PROC_BODY) V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_2_STREAMS.RU (PROC_BODY)             Size: 6
    Server:
    SAFE_STREAMS.MERGE_2_STREAMS.IRU1 (PROC_SPEC)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    SAFE_STREAMS.MERGE_2_STREAMS.IRU2 (PROC_SPEC)           V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_2_STREAMS.X_ADVANCE (PROC_BODY)      Size: 41
    Server:
    GLOBAL_TYPES (PAC_SPEC)                                 V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
    SAFE_STREAMS.MERGE_2_STREAMS.IA1 (PROC_SPEC)            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    SAFE_STREAMS.MERGE_2_STREAMS.IA2 (PROC_SPEC)            V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_2_STREAMS.X_REGISTER_USER (PROC_BODY)Size: 24
    Server:
    LIST_MANAGER (PAC_SPEC)                                 V 0 G 0 0 1 C 4 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                    V 0 G 0 0 1 C 4 :: V 0 G 0 0 1 C 0
    LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)             V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)              V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)         V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_2_STREAMS.X_CLOSE_STREAM (PROC_BODY) Size: 11
```

```
    Server:
    LIST_MANAGER (PAC_SPEC)                           V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)              V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)   V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_2_STREAMS.X_OPEN_STREAM (PROC_BODY)  Size: 11
  * SAFE_STREAMS.MERGE_2_STREAMS.ADVANCE (PROC_BODY)        Size: 7
    Server:
    SAFE_STREAMS.MERGE_2_STREAMS.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1


  * SAFE_STREAMS.MERGE_2_STREAMS.REGISTER_USER (PROC_BODY)  Size: 6
    Server:
    SAFE_STREAMS.MERGE_2_STREAMS.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_2_STREAMS.OPEN_STREAM (PROC_BODY)    Size: 6
    Server:
    SAFE_STREAMS.MERGE_2_STREAMS.STREAM_GUARD.OPEN_STREAM (ENTRY_SPEC)
                                                      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_2_STREAMS.CLOSE_STREAM (PROC_BODY)   Size: 4
    Server:
    SAFE_STREAMS.MERGE_2_STREAMS.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.MERGE_2_STREAMS.STREAM_GUARD (TASK_BODY)   Size: 14
* SAFE_STREAMS.CONCAT_2_STREAMS (PAC_BODY)                  Size: 172
  Server:
  LIST_MANAGER (PAC_SPEC)                             V 0 G 1 0 2 C 12 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                V 0 G 1 0 2 C 12 :: V 0 G 0 0 1 C 0
  LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)          V 0 G 0 0 0 C 4 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)     V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)         V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)       V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)    V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
  * SAFE_STREAMS.CONCAT_2_STREAMS.CUSER_HANDLE_LIST (GEN_PAC_INS)
                                                      Size: 1
    Gen_Ins     : LIST_MANAGER.LIST_TYPE
    Server:
    LIST_MANAGER (PAC_SPEC)                           V 0 G 1 0 0 C 0 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)              V 0 G 1 0 0 C 0 :: V 0 G 1 0 0 C 0
  * SAFE_STREAMS.CONCAT_2_STREAMS.STREAM_GUARD (TASK_TYP)   Size: 12
    * SAFE_STREAMS.CONCAT_2_STREAMS.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                      Size: 4
      Client:
      SAFE_STREAMS.CONCAT_2_STREAMS.ADVANCE (PROC_BODY)     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.CONCAT_2_STREAMS.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                      Size: 3
      Client:
      SAFE_STREAMS.CONCAT_2_STREAMS.REGISTER_USER (PROC_BODY)
                                                      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.CONCAT_2_STREAMS.STREAM_GUARD.OPEN_STREAM (ENTRY_SPEC)
                                                      Size: 3
      Client:
      SAFE_STREAMS.CONCAT_2_STREAMS.OPEN_STREAM (PROC_BODY) V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.CONCAT_2_STREAMS.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                      Size: 1
      Client:
      SAFE_STREAMS.CONCAT_2_STREAMS.CLOSE_STREAM (PROC_BODY)V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_2_STREAMS.SET_UP_NEXT_STREAM (PROC_BODY)
                                                      Size: 11
    Server:
    LIST_MANAGER (PAC_SPEC)                           V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)              V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    SAFE_STREAMS.CONCAT_2_STREAMS.IRU1 (PROC_SPEC)    V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    SAFE_STREAMS.CONCAT_2_STREAMS.IRU2 (PROC_SPEC)    V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_2_STREAMS.OK_TO_SET_UP (FUNC_BODY)  Size: 10
```

```
    Server:
    LIST_MANAGER (PAC_SPEC)                             V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                V 0 G 0 0 0 C 3 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.RESET_NEXT (PROC_SPEC)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM (PROC_SPEC)    V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1

  * SAFE_STREAMS.CONCAT_2_STREAMS.X_ADVANCE (PROC_BODY)   Size: 35
    Server:
    SAFE_STREAMS.CONCAT_2_STREAMS.IA1 (PROC_SPEC)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    SAFE_STREAMS.CONCAT_2_STREAMS.IA2 (PROC_SPEC)       V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_2_STREAMS.X_REGISTER_USER (PROC_BODY)
                                                        Size: 28
    Server:
    LIST_MANAGER (PAC_SPEC)                             V 0 G 0 0 1 C 4 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                V 0 G 0 0 1 C 4 :: V 0 G 0 0 1 C 0
    LIST_MANAGER.LIST_TYPE.ADD_ELEM (FUNC_SPEC)         V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
    SAFE_STREAMS.CONCAT_2_STREAMS.IRU1 (PROC_SPEC)      V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
    SAFE_STREAMS.CONCAT_2_STREAMS.IRU2 (PROC_SPEC)      V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 2
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_2_STREAMS.X_CLOSE_STREAM (PROC_BODY)Size: 13
    Server:
    LIST_MANAGER (PAC_SPEC)                             V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE (GENERIC_PAC)                V 0 G 0 0 0 C 2 :: V 0 G 0 0 0 C 0
    LIST_MANAGER.LIST_TYPE.SIZE_OF (FUNC_SPEC)          V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    LIST_MANAGER.LIST_TYPE.REMOVE_FIRST (PROC_SPEC)     V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    SAFE_STREAMS.CONCAT_2_STREAMS.IRU1 (PROC_SPEC)      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    SAFE_STREAMS.CONCAT_2_STREAMS.IRU2 (PROC_SPEC)      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_2_STREAMS.X_OPEN_STREAM (PROC_BODY) Size: 10
  * SAFE_STREAMS.CONCAT_2_STREAMS.ADVANCE (PROC_BODY)     Size: 7
    Server:
    SAFE_STREAMS.CONCAT_2_STREAMS.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_2_STREAMS.REGISTER_USER (PROC_BODY) Size: 6
    Server:
    SAFE_STREAMS.CONCAT_2_STREAMS.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_2_STREAMS.OPEN_STREAM (PROC_BODY)  Size: 6
    Server:
    SAFE_STREAMS.CONCAT_2_STREAMS.STREAM_GUARD.OPEN_STREAM (ENTRY_SPEC)
                                                        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_2_STREAMS.CLOSE_STREAM (PROC_BODY)  Size: 4
    Server:
    SAFE_STREAMS.CONCAT_2_STREAMS.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                        V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
  * SAFE_STREAMS.CONCAT_2_STREAMS.STREAM_GUARD (TASK_BODY)  Size: 14
* SAFE_STREAMS.OUTPUT_DRIVER (PROC_BODY)                Size: 17
  Server:
  GLOBAL_TYPES (PAC_SPEC)                               V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
* SAFE_STREAMS.REDUCE_STREAM (PROC_BODY)                Size: 16
  Server:
  GLOBAL_TYPES (PAC_SPEC)                               V 1 G 0 0 0 C 0 :: V 1 G 0 0 0 C 0
  Ctxt:
  LIST_MANAGER
  DIRECT_IO
  GLOBAL_TYPES
* SAFE_STREAMS.ENUM_STREAM (PAC_BODY)                   Size: 94
  * SAFE_STREAMS.ENUM_STREAM.STREAM_GUARD (TASK_TYP)      Size: 12
    * SAFE_STREAMS.ENUM_STREAM.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                        Size: 4
      Client:
      SAFE_STREAMS.ENUM_STREAM.ADVANCE (PROC_BODY)      V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1
    * SAFE_STREAMS.ENUM_STREAM.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                        Size: 3
      Client:
      SAFE_STREAMS.ENUM_STREAM.REGISTER_USER (PROC_BODY)  V 0 G 0 0 0 C 1 :: V 0 G 0 0 0 C 1

    * SAFE_STREAMS.ENUM_STREAM.STREAM_GUARD.OPEN_STREAM (ENTRY_SPEC)
```

```
                                                           Size: 3
     Client:
     SAFE_STREAMS.ENUM_STREAM.OPEN_STREAM (PROC_BODY)      V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
   * SAFE_STREAMS.ENUM_STREAM.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                           Size: 1
     Client:
     SAFE_STREAMS.ENUM_STREAM.CLOSE_STREAM (PROC_BODY)     V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
  * SAFE_STREAMS.ENUM_STREAM.X_ADVANCE (PROC_BODY)         Size: 18
  * SAFE_STREAMS.ENUM_STREAM.X_REGISTER_USER (PROC_BODY)   Size: 11
  * SAFE_STREAMS.ENUM_STREAM.X_OPEN_STREAM (PROC_BODY)     Size: 8
  * SAFE_STREAMS.ENUM_STREAM.X_CLOSE_STREAM (PROC_BODY)    Size: 7
  * SAFE_STREAMS.ENUM_STREAM.ADVANCE (PROC_BODY)           Size: 7
    Server:
    SAFE_STREAMS.ENUM_STREAM.STREAM_GUARD.ADVANCE (ENTRY_SPEC)
                                                           V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
  * SAFE_STREAMS.ENUM_STREAM.REGISTER_USER (PROC_BODY)     Size: 6
    Server:
    SAFE_STREAMS.ENUM_STREAM.STREAM_GUARD.REGISTER_USER (ENTRY_SPEC)
                                                           V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
  * SAFE_STREAMS.ENUM_STREAM.OPEN_STREAM (PROC_BODY)       Size: 6
    Server:
    SAFE_STREAMS.ENUM_STREAM.STREAM_GUARD.OPEN_STREAM (ENTRY_SPEC)
                                                           V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
  * SAFE_STREAMS.ENUM_STREAM.CLOSE_STREAM (PROC_BODY)      Size: 4
    Server:
    SAFE_STREAMS.ENUM_STREAM.STREAM_GUARD.CLOSE_STREAM (ENTRY_SPEC)
                                                           V O G 0 0 0 C 1 :: V O G 0 0 0 C 1
  * SAFE_STREAMS.ENUM_STREAM.STREAM_GUARD (TASK_BODY)      Size: 14
* GLOBAL_TYPES (PAC_BODY)                                  Size: 2
```

C Reuse Report At Top Level For Data Set - 1

```
CLIENT_PERSPECTIVE

 1. IO_EXCEPTIONS (Interface) Size: 9
 2. SEQUENTIAL_IO (Interface) Size: 36 Gen_Size: 1
    1. IO_EXCEPTIONS(PAC_SPEC)                               7  0  0   0 ::  7  0  0  0    9
 3. DIRECT_IO (Interface) Size: 48 Gen_Size: 1
    1. IO_EXCEPTIONS(PAC_SPEC)                               7  0  0   0 ::  7  0  0  0    9
 4. TEXT_IO (Interface) Size: 195
    1. IO_EXCEPTIONS(PAC_SPEC)                               8  0  0   0 ::  8  0  0  0    9
 5. MACHINE_CODE (Interface) Size: 279
 6. UNCHECKED_CONVERSION (Interface) Size: 3 Gen_Size: 2
 7. UNCHECKED_DEALLOCATION (Interface) Size: 3 Gen_Size: 2
 8. LOW_LEVEL_IO (Interface) Size: 8
 9. CALENDAR (Interface) Size: 40
10. SYSTEM (Interface) Size: 49
11. GLOBAL_TYPES (Interface) Size: 5
12. LIST_MANAGER (Interface) Size: 23
    1. GLOBAL_TYPES(PAC_SPEC)                                2  0  0   0 ::  0  0  0  0    5
       * LIST_MANAGER (Body) Size: 90
         1. GLOBAL_TYPES(PAC_SPEC)                           1  0  0   0 ::  0  0  0  0    5
13. SAFE_STREAMS (Interface) Size: 290
    1. GLOBAL_TYPES(PAC_SPEC)                                4  0  0   0 ::  0  0  0  0    5
       *  SAFE_STREAMS (Body) Size: 1789
          1. GLOBAL_TYPES(PAC_SPEC)                         15  0  0   0 ::  0  0  0  0    5
          2. LIST_MANAGER(PAC_SPEC)                          0 14 24 128 ::  0  0  0  0    23
          3. LIST_MANAGER.LIST_TYPE(GENERIC_PAC)             0 14 24 128 ::  0  0  0  0    19
          4. LIST_MANAGER.LIST_TYPE.SIZE_OF(FUNC_SPEC)       0  0  0  37 ::  0  0  0  0    1
          5. LIST_MANAGER.LIST_TYPE.REMOVE_FIRST(PROC_SPEC)  0  0  0  25 ::  0  0  0  0    2
          6. LIST_MANAGER.LIST_TYPE.ADD_ELEM(FUNC_SPEC)      0  0  0  25 ::  0  0  0  0    2
          7. LIST_MANAGER.LIST_TYPE.RESET_NEXT(PROC_SPEC)    0  0  0  13 ::  0  0  0  0    1
          8. LIST_MANAGER.LIST_TYPE.GET_NEXT_ELEM(PROC_SPEC) 0  0  0  13 ::  0  0  0  0    2
          9. LIST_MANAGER.LIST_TYPE.NTH_ELEM(FUNC_SPEC)      0  0  0  15 ::  0  0  0  0    2
         10. DIRECT_IO(GENERIC_PAC)                          0  1  2   3 ::  0  0  0  0    48
         11. DIRECT_IO.READ(PROC_SPEC)                       0  0  0   1 ::  0  0  0  0    3
         12. DIRECT_IO.CLOSE(PROC_SPEC)                      0  0  0   1 ::  0  0  0  0    1
         13. DIRECT_IO.DELETE(PROC_SPEC)                     0  0  0   1 ::  0  0  0  0    1

************

SERVER_PERSPECTIVE
 1. IO_EXCEPTIONS (Interface) Size: 9
     1. SEQUENTIAL_IO(GENERIC_PAC)                           7  0  0   0 ::  7  0  0  0    36
     2. DIRECT_IO(GENERIC_PAC)                               7  0  0   0 ::  7  0  0  0    48
     3. TEXT_IO(PAC_SPEC)                                    8  0  0   0 ::  8  0  0  0    195
 2. SEQUENTIAL_IO (Interface) Size: 36 Gen_Size: 1
 3. DIRECT_IO (Interface) Size: 48 Gen_Size: 1
     1. SAFE_STREAMS(PAC_BODY)                               0  1  2   3 ::  0  0  0  0    149
     2. SAFE_STREAMS.FEED_STREAM.SIO(GEN_PAC_INS)            0  1  0   0 ::  0  1  0  0    1
     3. SAFE_STREAMS.FEED_STREAM(PAC_BODY)                   0  1  2   3 ::  0  0  2  0    268
     4. SAFE_STREAMS.FEED_STREAM.READ_NEXT(PROC_BODY)        0  0  0   1 ::  0  0  0  0    15
     5. SAFE_STREAMS.FEED_STREAM.CF(PROC_BODY)               0  0  0   2 ::  0  0  0  0    7
 4. TEXT_IO (Interface) Size: 195
 5. MACHINE_CODE (Interface) Size: 279
 6. UNCHECKED_CONVERSION (Interface) Size: 3 Gen_Size: 2
 7. UNCHECKED_DEALLOCATION (Interface) Size: 3 Gen_Size: 2
 8. LOW_LEVEL_IO (Interface) Size: 8
 9. CALENDAR (Interface) Size: 40
10. SYSTEM (Interface) Size: 49
11. GLOBAL_TYPES (Interface) Size: 5
    1. LIST_MANAGER(PAC_SPEC)                                2  0  0   0 ::  0  0  0  0    23
    2. LIST_MANAGER.LIST_TYPE(GENERIC_PAC)                   2  0  0   0 ::  0  0  0  0    19
    3. LIST_MANAGER.LIST_TYPE.SIZE_OF(FUNC_SPEC)             1  0  0   0 ::  1  0  0  0    1
    4. LIST_MANAGER.LIST_TYPE.NTH_ELEM(FUNC_SPEC)            1  0  0   0 ::  1  0  0  0    2
    5. LIST_MANAGER(PAC_BODY)                                1  0  0   0 ::  0  0  0  0    90
    6. LIST_MANAGER.LIST_TYPE(PAC_BODY)                      1  0  0   0 ::  1  0  0  0    89
    7. SAFE_STREAMS(PAC_SPEC)                                4  0  0   0 ::  0  0  0  0    290
```

```
     8. SAFE_STREAMS.FEED_STREAM(GENERIC_PAC)                  1  0  0   0 ::  1  0  0  0    23
     9. SAFE_STREAMS(PAC_BODY)                                15  0  0   0 ::  0  0  0  0    149
    10. SAFE_STREAMS.FEED_STREAM(PAC_BODY)                     2  0  0   0 ::  1  0  0  0    268
    11. SAFE_STREAMS.MERGE_STREAM(PAC_BODY)                    7  0  0   0 ::  1  0  0  0    181
    12. SAFE_STREAMS.GEN_STREAM(GENERIC_PAC)                   1  0  0   0 ::  1  0  0  0    17
    13. SAFE_STREAMS.GEN_STREAM(PAC_BODY)                      2  0  0   0 ::  1  0  0  0    177
    14. SAFE_STREAMS.MERGE_2_STREAMS(PAC_BODY)                 2  0  0   0 ::  1  0  0  0    157
    15. SAFE_STREAMS.REDUCE_STREAM(GENERIC_PROC)               1  0  0   0 ::  1  0  0  0    16
    16. SAFE_STREAMS.OUTPUT_DRIVER(GENERIC_PROC)               1  0  0   0 ::  1  0  0  0    16
    17. SAFE_STREAMS.OUTPUT_DRIVER(PROC_BODY)                  1  0  0   0 ::  1  0  0  0    17
    18. SAFE_STREAMS.REDUCE_STREAM(PROC_BODY)                  1  0  0   0 ::  1  0  0  0    16
    19. SAFE_STREAMS.MERGE_2_STREAMS.X_ADVANCE(PROC_BODY)      1  0  0   0 ::  1  0  0  0    41
    20. SAFE_STREAMS.GEN_STREAM.CHECK_BUFFER(PROC_BODY)        1  0  0   0 ::  1  0  0  0    18
    21. SAFE_STREAMS.MERGE_STREAM.BUMP(FUNC_BODY)              4  0  0   0 ::  4  0  0  0    7
    22. SAFE_STREAMS.MERGE_STREAM.X_ADVANCE(PROC_BODY)         2  0  0   0 ::  2  0  0  0    36
    23. SAFE_STREAMS.FEED_STREAM.CHECK_BUFFER(PROC_BODY)       1  0  0   0 ::  1  0  0  0    17
 12. LIST_MANAGER (Interface) Size: 23
     1. SAFE_STREAMS.FEED_STREAM.USER_HANDLE_LIST(GEN_PAC_INS)     0  1  0   0 ::  0  0  0  0    1
     2. SAFE_STREAMS.FEED_STREAM.STREAM_LIST(GEN_PAC_INS)      0  1  0   0 ::  0  0  0  0    1
     3. SAFE_STREAMS(PAC_BODY)                                 0 14 24 128 ::  0  0  0  0    149
     4. SAFE_STREAMS.FEED_STREAM(PAC_BODY)                     0  2  3  21 ::  0  0  0  0    268
     5. SAFE_STREAMS.MAP_STREAM.MUSER_HANDLE_LIST(GEN_PAC_INS)     0  1  0   0 ::  0  0  0  0    1
     6. SAFE_STREAMS.MAP_STREAM(PAC_BODY)                      0  1  2   6 ::  0  0  0  0    120
     7. SAFE_STREAMS.FILTER_STREAM.FUSER_HANDLE_LIST(GEN_PAC_INS)
                                                               0  1  0   0 ::  0  0  0  0    1
     8. SAFE_STREAMS.FILTER_STREAM(PAC_BODY)                   0  1  2   6 ::  0  0  0  0    124
     9. SAFE_STREAMS.MERGE_STREAM.MUSER_HANDLE_LIST(GEN_PAC_INS)   0  1  0   0 ::  0  0  0  0    1
    10. SAFE_STREAMS.MERGE_STREAM.ISTREAM_HANDLE_LIST(GEN_PAC_INS)
                                                               0  1  0   0 ::  0  0  0  0    1
    11. SAFE_STREAMS.MERGE_STREAM.IUSER_HANDLE_LIST(GEN_PAC_INS)
                                                               0  1  0   0 ::  0  0  0  0    1
    12. SAFE_STREAMS.MERGE_STREAM(PAC_BODY)                    0  3  4  26 ::  0  0  0  0    181
    13. SAFE_STREAMS.CONCAT_STREAM.CUSER_HANDLE_LIST(GEN_PAC_INS)
                                                               0  1  0   0 ::  0  0  0  0    1
    14. SAFE_STREAMS.CONCAT_STREAM.ISTREAM_HANDLE_LIST(GEN_PAC_INS)
                                                               0  1  0   0 ::  0  0  0  0    1
    15. SAFE_STREAMS.CONCAT_STREAM(PAC_BODY)                   0  2  3  21 ::  0  0  0  0    168
    16. SAFE_STREAMS.TRANS_STREAM.TUSER_HANDLE_LIST(GEN_PAC_INS)
                                                               0  1  0   0 ::  0  0  0  0    1
    17. SAFE_STREAMS.TRANS_STREAM(PAC_BODY)                    0  1  2  10 ::  0  0  0  0    146
    18. SAFE_STREAMS.GEN_STREAM.USER_HANDLE_LIST(GEN_PAC_INS)  0  1  0   0 ::  0  0  0  0    1
    19. SAFE_STREAMS.GEN_STREAM.STREAM_LIST(GEN_PAC_INS)       0  1  0   0 ::  0  0  0  0    1
    20. SAFE_STREAMS.GEN_STREAM(PAC_BODY)                      0  2  4  20 ::  0  0  0  0    177
    21. SAFE_STREAMS.MERGE_2_STREAMS.MUSER_HANDLE_LIST(GEN_PAC_INS)
                                                               0  1  0   0 ::  0  0  0  0    1
    22. SAFE_STREAMS.MERGE_2_STREAMS(PAC_BODY)                 0  1  2   6 ::  0  0  0  0    157
    23. SAFE_STREAMS.CONCAT_2_STREAMS.CUSER_HANDLE_LIST(GEN_PAC_INS)
                                                               0  1  0   0 ::  0  0  0  0    1
    24. SAFE_STREAMS.CONCAT_2_STREAMS(PAC_BODY)                0  1  2  12 ::  0  0  0  0    172
    25. SAFE_STREAMS.CONCAT_2_STREAMS.X_REGISTER_USER(PROC_BODY)
                                                               0  0  1   4 ::  0  0  0  0    28
    26. SAFE_STREAMS.MERGE_2_STREAMS.X_REGISTER_USER(PROC_BODY)    0  0  1   4 ::  0  0  0  0    24
    27. SAFE_STREAMS.GEN_STREAM.CHECK_BUFFER(PROC_BODY)        0  0  1   8 ::  0  0  0  0    18
    28. SAFE_STREAMS.GEN_STREAM.X_REGISTER_USER(PROC_BODY)     0  0  1   4 ::  0  0  0  0    22
    29. SAFE_STREAMS.CONCAT_STREAM.X_REGISTER_USER(PROC_BODY)  0  0  1   6 ::  0  0  0  0    27
    30. SAFE_STREAMS.TRANS_STREAM.X_REGISTER_USER(PROC_BODY)   0  0  1   4 ::  0  0  0  0    22
    31. SAFE_STREAMS.MERGE_STREAM.X_REGISTER_USER(PROC_BODY)   0  0  1   4 ::  0  0  0  0    24
    32. SAFE_STREAMS.FILTER_STREAM.X_REGISTER_USER(PROC_BODY)  0  0  1   4 ::  0  0  0  0    24
    33. SAFE_STREAMS.MAP_STREAM.X_REGISTER_USER(PROC_BODY)     0  0  1   4 ::  0  0  0  0    24
    34. SAFE_STREAMS.FEED_STREAM.X_REGISTER_USER(PROC_BODY)    0  0  1   4 ::  0  0  0  0    22
    35. SAFE_STREAMS.CONCAT_2_STREAMS.X_CLOSE_STREAM(PROC_BODY)    0  0  0   2 ::  0  0  0  0    13
    36. SAFE_STREAMS.MERGE_2_STREAMS.X_CLOSE_STREAM(PROC_BODY)     0  0  0   2 ::  0  0  0  0    11
    37. SAFE_STREAMS.GEN_STREAM.X_CLOSE_STREAM(PROC_BODY)      0  0  0   4 ::  0  0  0  0    14
    38. SAFE_STREAMS.CONCAT_STREAM.SET_UP_NEXT_STREAM(PROC_BODY)
                                                               0  0  0   5 ::  0  0  0  0    14
    39. SAFE_STREAMS.CONCAT_STREAM.X_CLOSE_STREAM(PROC_BODY)   0  0  0   5 ::  0  0  0  0    15
    40. SAFE_STREAMS.MERGE_STREAM.DRU(PROC_BODY)               0  0  0   4 ::  0  0  0  0    10
    41. SAFE_STREAMS.MERGE_STREAM.X_CLOSE_STREAM(PROC_BODY)    0  0  0   4 ::  0  0  0  0    14
```

```
   42. SAFE_STREAMS.FILTER_STREAM.X_CLOSE_STREAM(PROC_BODY)        0  0  0   2 ::  0  0  0  0    11
   43. SAFE_STREAMS.MAP_STREAM.X_CLOSE_STREAM(PROC_BODY)           0  0  0   2 ::  0  0  0  0    11
   44. SAFE_STREAMS.FEED_STREAM.CHECK_BUFFER(PROC_BODY)            0  0  0   8 ::  0  0  0  0    17
   45. SAFE_STREAMS.FEED_STREAM.X_CLOSE_STREAM(PROC_BODY)          0  0  0   4 ::  0  0  0  0    14
   46. SAFE_STREAMS.GEN_STREAM.X_ADVANCE(PROC_BODY)                0  0  0   3 ::  0  0  0  0    36
   47. SAFE_STREAMS.CONCAT_STREAM.X_ADVANCE(PROC_BODY)             0  0  0   1 ::  0  0  0  0    32
   48. SAFE_STREAMS.MERGE_STREAM.X_ADVANCE(PROC_BODY)              0  0  0   5 ::  0  0  0  0    36
   49. SAFE_STREAMS.FEED_STREAM.X_ADVANCE(PROC_BODY)               0  0  0   3 ::  0  0  0  0    36
   50. SAFE_STREAMS.CONCAT_2_STREAMS.SET_UP_NEXT_STREAM(PROC_BODY)
                                                                   0  0  0   3 ::  0  0  0  0    11
   51. SAFE_STREAMS.CONCAT_2_STREAMS.OK_TO_SET_UP(FUNC_BODY)       0  0  0   3 ::  0  0  0  0    10
   52. SAFE_STREAMS.CONCAT_STREAM.OK_TO_SET_UP(FUNC_BODY)          0  0  0   3 ::  0  0  0  0    10
   53. SAFE_STREAMS.TRANS_STREAM.CHECK_USERS(FUNC_BODY)            0  0  0   3 ::  0  0  0  0    8
   54. SAFE_STREAMS.TRANS_STREAM.RESET_USERS(PROC_BODY)            0  0  0   3 ::  0  0  0  0    7
   55. SAFE_STREAMS.MERGE_STREAM.RU(PROC_BODY)                     0  0  0   4 ::  0  0  0  0    10
   56. SAFE_STREAMS.MERGE_STREAM.X_MERGE(PROC_BODY)                0  0  0   5 ::  0  0  0  0    18
   57. SAFE_STREAMS.GEN_STREAM.READ_NEXT(PROC_BODY)                0  0  0   1 ::  0  0  0  0    10
   58. SAFE_STREAMS.CONCAT_STREAM.X_CONCAT(PROC_BODY)              0  0  0   1 ::  0  0  0  0    9
   59. SAFE_STREAMS.FEED_STREAM.READ_NEXT(PROC_BODY)               0  0  0   1 ::  0  0  0  0    15
   60. SAFE_STREAMS.FEED_STREAM.X_FEED(PROC_BODY)                  0  0  0   1 ::  0  0  0  0    14
       * LIST_MANAGER (Body) Size: 90
13. SAFE_STREAMS (Interface) Size: 290
       * SAFE_STREAMS (Body) Size: 1789

************
```