Pascal-F Verifier Internal Design Document

Design Overview

John Nagle
Scott Johnson

FACC / Palo Alto

1. Overall System Design

1.1 CPCI #1 -- Language Processing

1.1.1 Phase 1A

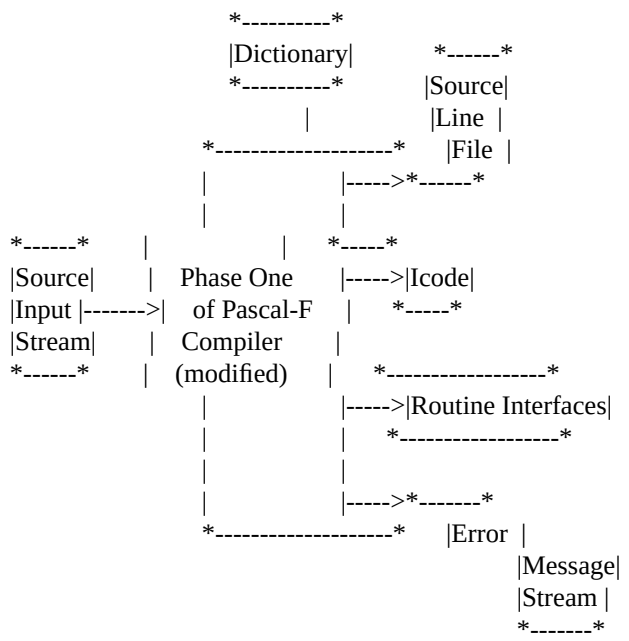

Figure 1. Phase 1A -- Syntax and Type Checking

The syntax and type checking phase has the task of accepting Pascal-F input and producing a representation of the program in a reverse Polish notation called ‘‘Icode’’. This phase is also required to perform the normal checks expected of a Pascal compiler, including syntax checks and variable type checks.

Phase 1 of the Pascal-F compiler will be modified to accept the additional language constructs required for verification and will then become phase 1 of both the compiler and the veriifer.

Inputs and outputs of this phase are as follows:

Source Input	An ASCII text file containing the source program. Read only.
Source Lines	A file of source lines indexed in such a way that any line can be retrieved by line number. This file exists to allow retrieval of program text for use in diagnostic messages of later phases. Write only in this phase.

Dictionary	A file of named objects, indexed by name and scope, containing the properties of the objects. This file is a representation of the compiler's dictionary. Created in this phase.
Routine Database	A file of routines and properties thereof. This information includes • The list of procedures and functions directly called by each procedure and function • The list of imported variables directly referenced by each procedure and function. • The list of imported variables directly altered by each procedure and function. • The entry and exit assertions of each procedure and function. • The invariants of each module and monitor. • The priority of each procedure, function, monitor, and module. • The containing monitor or module of each procedure, function, monitor, and module. • For each procedure and function, answers to the questions: 1. Does the routine perform a ''wait'' operation? 2. Does the routine perform a ''send'' operation? This list may not be complete. Because the final definition of the Pascal-F language has not been received, it is not yet possible to be sure that all required information about routines is listed above. Created in this phase.
''Icode''	The reverse Polish form of the program text. A sequential file, readable forward only. Created in this phase.
Error Messages	User-readable error messages. Standard output.

1.2 CPCI #2: Preverification Checking and Jcode Generation

1.2.1 Phase 2A -- Transitive Property Propagation

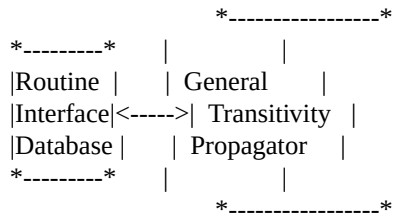

Figure 2. Transitive Property Propagation Phase

The task of this phase is to perform transitive closure on those properties in the Routine Interface Database which are transitive. For example, the knowledge that "procedure A calls procedure B" and "procedure B modifies variable X" would be combined to produce the information that "procedure A modifies variable X". This new information would then be added to the Routine Interface Database. The following properties are considered transitive: (This list may not be complete).

1. A calls B
2. A modifies X
3. A references X
4. A contains a ‘‘wait’’
5. A contains a ‘‘send’’

Inputs and outputs of this phase are as follows:

Interface Database	Read/write in this phase.

1.2.2 Phase 2B -- Jcode Generation

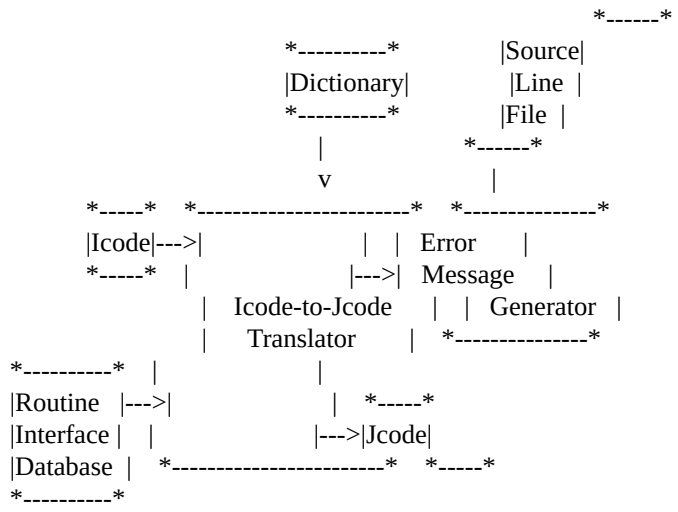

Figure 3. Jcode Generation

Jcode is a representation of the source program designed to accomodate the needs of the verification condition generator. Jcode is defined by the ‘‘Jcode Interface Control Document’’.

Icode contains addresses of variables; Jcode contains their names. The names of variables are obtained by searching an internal table of variables ordered by address, which is created from the dictionary.

Inputs and outputs of this phase are as follows:

Icode file	Read only in this phase.
Dictionary	Used to obtain definitions of variables and records. Read only in this phase.
Source Lines	Read only in this phase.
Interface Database	Read only in this phase.
Jcode	Created in this phase.
Error messages	Standard output.

1.3 CPCI #3: Verification Condition Generation

1.3.1 Phase 3A -- Jcode Augmentation

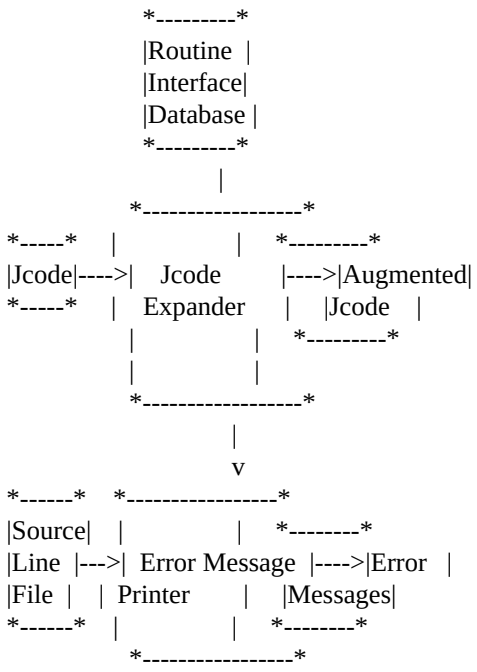

Figure 4. Phase 3A -- Jcode Augmentation

The primary purpose of Jcode augmentation is the expansion of procedure and function calls into assertions about the variables involved in the calls. The description of a routine in augmented Jcode stands alone; no information external to the Jcode is required to define the information to be proven about the routine. If the augmented Jcode of a routine does not change from one verification attempt to the next, there is no need to reverify the routine.

1.3.2 Phase 3B -- Verification Condition Generation

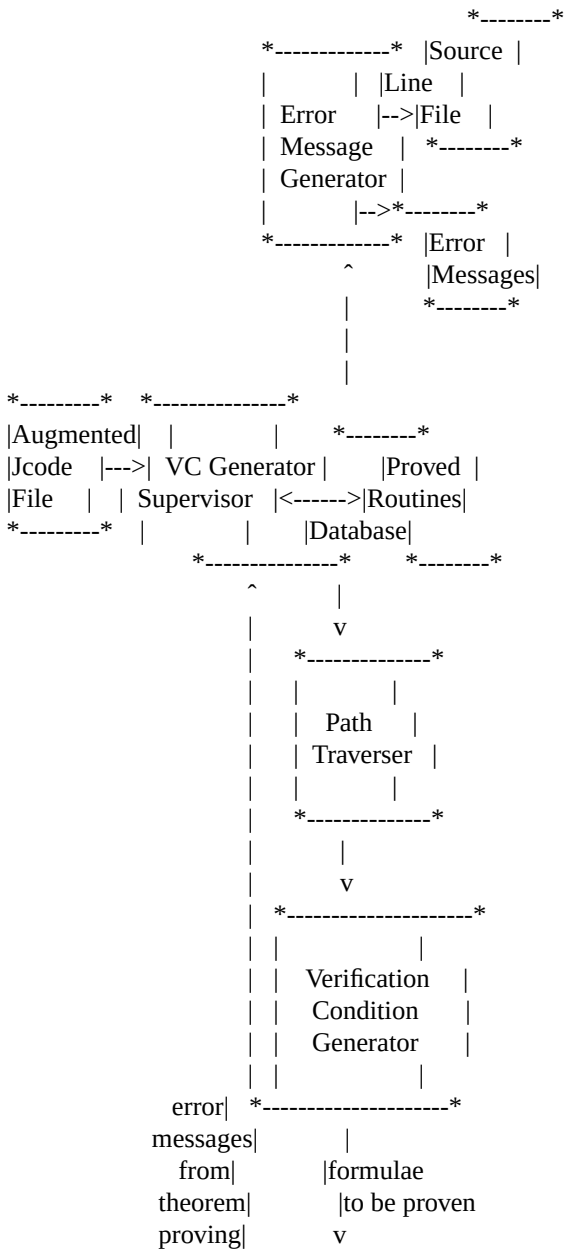

Figure 5. Phase 3B - Verification Condition Generation

It is the task of this phase to generate the verification conditions for which proof will be attempted. This phase interacts directly with the next phase, in that each verification condition, as it is generated, is submitted to the theorem prover, so that the success or failure of the proof is immediately available to the verification condition generator supervisor.

Inputs and outputs of this phase are as follows:

Jcode file	Read only in this phase.
Proved Routines	This database is used to avoid reverification of unchanged routines. If the augmented Jcode for a routine has not changed since the last verification attempt, verification condition generation and theorem proving are ommitted.
	The error messages generated during theorem proving are saved in this

database, so that reverification attempts produce the same error messages as previous tries.

Source Line File	Read-only in this phase.
Error Messages	Standard output.

1.4 CPCI #4: Theorem Proving

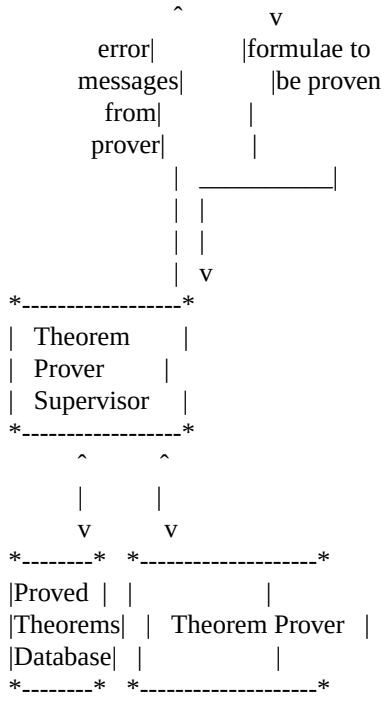

Figure 6. Theorem Proving Phase

Inputs and outputs of this phase are as follows:

VC file	Read only in this phase.
Proved Theorems	A dictionary of proved theorems used to avoid resubmitting already-proved (or disproved) conjectures to the theorem prover. This database is preserved from run to run of the verifier as a file associated with the program being verified. Read/write in this phase; may be created empty if nonexistent.
Error Stream	Error messages. Write only.