Gardens Point Component Pascal — Change Log

John Gough

March 29, 2018

This document applies to GPCP version v1.4.07.

1 About Gardens Point Component Pascal (*gpcp*)

Gardens Point Component Pascal (*gpcp*) is an implementation of the *Component Pascal* Language, as defined in the *Component Pascal Report* from Oberon Microsystems. It is intended that this be a faithful implementation of the Report, except for those differences that are explicitly detailed in the Release Notes.

The distribution consists of four programs, and a number of libraries. The programs are the compiler *gpcp*, the make utility *CPMake*, a module interface browser tool *Browse*, and a tool for extracting public symbol metadata from assemblies written in other languages. For the *JVM* version this tool is named *J2cps*; for the *.NET* version the tool is named *PeToCps*.

The compiler produces either *.NET* Common Intermediate Language (*CIL*) or *Java* byte-codes as output. The compiler can be bootstrapped on either platform.

2 Tracking the Changes

Where to get *gpcp*

Updates are announced and available from `https://github.com/k-john-gough/gpcp`

How to Report Bugs

If you find what you believe is a bug, please file an issue on the project page with the detail of the event. It would be particularly helpful if you can send the code of the shortest program which can illustrate the error.

Posting to the Mail Group

There is a discussion group for users of *gpcp*. You may subscribe by sending an email to GPCP-subscribe@yahoogroups.com. The development team monitor traffic on the group, and post update messages there.

3 Change summary – v1.4.07

Changes from 1.4.06

Changes in these version 1.4.07 releases apply to both targets. They provide better facilities for accessing functionality from the *Java* and *.NET* standard libraries. The previous limitations were discovered during the development of the forthcoming *PE*-file emitter based on the *System.Reflection.Emit* module.

There is a significant change to the *Browse* tool, with additional information on the limitations of the interfaces to foreign libraries. See the *Module Interface Browser* section of the release notes.

There is one change which may be a breaking change for some users.

- ⋆ Prior to version 1.4.07 different implementations of *GPBinFiles.readByte* behaved inconsistently. If an attempt was made to read past the end of file the *JVM* version threw an exception, while the *.NET* version returned -1. To have programs portable between implementations this difference has to be removed even though this will be a breaking change for some users. From version 1.4.07 both versions will return -1 if an attempt is made to read past the end of file.
- ⋆ Http output from the *Browse* tool now includes comments which, for foreign language symbol-files, spell out any limitation on the semantics of each defined type. These comments have pop-up explanations when clicked with a pointing device.
- ⋆ A bug in the implementation of the `/jasmin` option which caused an assert trap on modules with *WITH* statements has been removed.
- ⋆ Previous versions of *gpcp* were incorrectly rejecting value-assignments of *.NET* `structs`. In fact such assignments are permitted by the *.NET* runtime, and are now available for version 1.4.07+.
- ⋆ The *JVM* distribution now includes both the class-file trees for the tool executables, as well as the executable jar-files. Thus *gpcp*, for example, may be invoked by the command `gpcp` *args*, launching `gpcp.jar`, or by the command `cprun gpcp` *args*, launching `CP/gpcp/gpcp.class`.
- ⋆ A new option `/rflemit` has been introduced. This will be activated when the Reflection-Emit filewriter is finished.

Changes from 1.4.05

Changes in these version 1.4.06 releases apply mainly to the *.NET* target, but there are also some bug fixes to the *JVM* version.

The major change is the removal of the dependency on *PERWAPI* which will now be removed from the distribution. *PeToCps* now depends on *System.Reflection* to create symbol files from *PE*-files. The program should now be robust against changes in executable file format.

- ⋆ A new version of *PeToCps* is included. This has made a change to the options for this program. Most significantly the processing of `mscorlib` requires use of a new option. It is also only able to create a symbol file for the currently executing version of `mscorlib`. Behaviour for other libraries is unchanged.

- ⋆ The new version of *PeToCps* filters the list of features declared on abstract types, such as `System.Char`, and their companion primitive types, such as `char`. The symbol file includes all the static fields and static methods but excludes the instance features. Instance features on primitive types are, in any case, inaccessible to *Component Pascal*.

- ⋆ The *Browse* program now sorts the import list when the `/html` option is in force. For record types with long implements-lists the link to typebound procedures is reformatted to avoid being pushed off-screen.

- ⋆ There was an error in the encoding of the record-type-specific value copy function, `__copy__`, if the record contained a fixed-length array field that was not the final field of the record. This error had been there since version 0.5. Now fixed.

- ⋆ In the *JVM* version, for SE 7+, if the end of a module body was not reachable in the control flow graph the emission of a "`return`" byte-code caused a verification error. Now fixed.

- ⋆ In the *JVM* version, if a procedure body was empty, but enclosed one or more nested procedures *gpcp* crashed while trying to compute the data flow in the (inaccessible) nested procedure(s). This bug is common to both the `-legacy` and `-jasmin` code emitters. *gpcp* now processes the parameter list of the enclosing procedure, fixing the bug.

Changes from 1.4.04

All of the host-independent bug-fixes of version 1.4.04 have been applied to v1.4.05 of *gpcp-NET*. The following additional changes are applied in *gpcp-NET*.

- ⋆ There is a new utility *MakeIndex* that creates an index file "`index.html`" in a symbol-browse directory.

- ⋆ There is a new procedure in *ProgArgs*, `ExpandWildcards`. This is applied in both `gpcp` and `Browse`. Previously only the *JVM* host allowed commands like "`gpcp *.cp`".

- ⋆ The compiler no longer offers the `perwapi` option. `perwapi.dll` is still included in the distribution because *PeToCps* still requires it. The next version will instead use `system.reflection.emit`.

- ⋆ All the program examples have been put back into the distribution under the *documentation* directory. A readme file distinguishes files that are *.NET*-only or *JVM*-only, or are platform-agnostic.

The following *gpcp-JVM*-related bug fixes have been pushed to the source tree, but are not in v1.4.04 *gpcp-JVM*.

- ⋆ Fixing unimplemented method trap in *CallGetClass* of `AsmUtil.cp`

- ⋆ Fix option parsing of "/hsize=" vs "-hsize:" that broke on MS command line.

- ⋆ Fixed import-order dependent failure when importing `RTS` and `java_lang`.

Changes from 1.3.16

The following major changes have been made to the v1.4.04 JVM version of *gpcp*.

- The packaging of the compiler has been modified so that the compiler, and the other tools, may be invoked from the *jar* files included in the distribution. However, the default compiler output is unchanged, as a class-file hierarchy.
- The *JVM* version now uses the *ASM v5* library to emit its class files. This enables the compiler to emit files that are compatible with recent versions of the *Java SE* platform.
- New command line flags allow the compiler to produce class files for various *JVM* class-file standards. Output targeting *SE 6+* uses the new, preferred verification framework. This slows down compilation slightly but results in faster class loading.
- The previous direct class file writer is still included in the distribution, invoked by using the `-legacy` option. This file-writer will be retired at some stage in the future, but is included in v1.4, so that *gpcp-JVM* may still be cross-compiled using *gpcp-NET*.
- A very substantial rewrite of the jave-to-symbolfile utility is now part of the distribution. This new version no longer fails when dealing with recent library jar-files. Furthermore this program deals directly with “`jar`” files as well as class-file file-trees.
- A number of errors, have been fixed, removing a number of crashes inside the compiler for invalid source files.
- Use of pointer-type local variables that are not *definitely assigned* previously raised a warning – ”local variable may have its default NIL value”. This is now a fatal error, as it leads to generation of *Java* bytecodes that fail verification.
- *Browse* has also had a significant rewrite. It now works corretly when given multiple symbol files, including the common case of “`browse *.cps`”.

The corresponding changes to the *.NET* version will arrive later this year.

Changes from 1.3.15

The following corrections and changes are included in the 1.3.16 release.

- Fixed a bug with builtin arithmetic shift function *ASH* when applied to 64-bit operands.
- Added new builtin logical shift function *LSH*. This function applies to 32 and 64-bit integers. As for the standard *ASH* function, positive shifts are leftward.
- The semantics of both shift operations have been changed. Shift amount is now range checked, and shifts of greater than or equal to data-word width return zero or minus one as required.
- Added new builtin rotate function *ROT*. This generic function can rotate any integer-typed value from 8 to 64-bits. As with the shift functions, positive shifts are leftward.

- Fixed a bug with anonymous return types of procedures. Thus, public procedures may return (pointers to) anonymous arrays of public types without error.
- Corrected an error with anonymous procedure types on the *JVM* version.
- The IL emitter of the *.NET* version now uses the invariant culture to write *REAL* literal values. This fixes an issue for host machines with non-Anglocentric localization settings.

Changes from 1.3.14

The following change is included in the 1.3.15 release.

- The prohibition on writing to the guarded variable within a *WITH* statement has been varied to make it compatible with the behavior of *BlackBox Component Builder*. If the guarded variable is of *record* type it is now allowed to write to the fields. However any attempt to change the type of the guarded variable is a semantic error.
- The behavior with *pointer* types is unchanged. The fields of the object may be written to, but the pointer itself is read-only.

Changes from 1.3.13

The following corrections and changes are included in the 1.3.14 release.

- Procedure Types and variables are now supported for the *JVM* target, with the same limitation as for the *.NET* target. Specifically, values of procedure type are compatible if the types have the same *name*. The *Report* requires that values with the same *signature* be compatible.
- A error in the generation of the value copy runtime support methods for the *JVM* target has been corrected. The error was rather obscure, but caused some permitted entire assignments to fail to copy some base-class fields under certain specific circumstances.

Changes from 1.3.12

The following corrections and changes are included in the 1.3.13 release.

- A significant rewrite of *J2CPS* has corrected a bug. The bug caused a module import to be missed under certain very specific, rare circumstances.
- *gpcp* now populates the definition of *RTS.NativeObject* with the appropriate methods from the underlying platform base type, *java.lang.Object* or or *System.Object*, depending on the target platform setting. This means that, for example, a type derived from *RTS.NativeObject* may override these methods without an explicit import of the whole of the system module.

Changes from 1.3.11

The following corrections and changes are included in the 1.3.12 release.

- Symbol file reading and writing have been modified so that string literals may include arbitrary Unicode, and be of unbounded size.
- Literal handling throughout the compiler has been rewritten to allow for the possibility that strings might contain embedded NUL characters.
- Reading and writing of Unicode character sequences in symbol files now uses *modified* UTF-8.
- A new pseudo-module import *STA* causes the compiler to emit a code wrapper that runs the module body in a new thread with the STA property set to true.
- Some significant errors in the implementation of vectors of *CHAR* element type have been corrected, as has some inconsistency in the implementation of entire assignment for the vector types.
- Some programs that imported both *RTS* and mscorlib were were finding that native string receivers were being denied access to the inherited methods of *System.Object*. This is now fixed.
- Programs using the *TYPEOF* extension function now work correctly when compiled with *PERWAPI*.

One consequence of these changes is that conversions between character arrays, string literals and open arrays of characters have been made consistent with the language standard. This might constitute a breaking change for programs that were relying on an implicit "stringification" of an argument array. Within the code of the compiler there was exactly one case where a call that passed an argument array "`arr`" had to be changed to the correct, "`arr$`", form.

Changes from 1.3.10

The following corrections and changes are included in the 1.3.11 release.

- A new diagnostic message is added for unresolved opaque types when using the /perwapi option.
- When an opaque type is unresolved due to a missing import the PEFile Writer attempts to correct the situation by generating a dummy import and a corresponding type-ref descriptor.

Changes from 1.3.9

The following corrections and changes are included in the 1.3.10 release.

- The resolution of calls to overloaded methods from foreign language libraries now takes place in two steps. First an exact match of the argument types to the method formals is attempted, followed, if necessary, by a match which uses no type conversions other than between the *Component Pascal* character array types and the platform string type, and between the *Component Pascal ANYREC* and *ANYPTR* types and the platform object type.

- ⋆ The dummy symbol files produced by *PeToCps* from *PE*-files now ignores non-CLS compliant methods that *Component Pascal* cannot call.
- ⋆ Some corrections to the encoding of the "vector types" extension avoid verifier objections.
- ⋆ Some corrections to code of the separate *PERWAPI* project avoid certain failures of *PeToCps*.
- ⋆ *PeToCps* does not create version information in symbol files for *PE*-files that are versioned but not strongly named.
- ⋆ *gpcp* now produces code for procedures with covariant return types that is verifiable.
- ⋆ The *RealStr* library now uses the invariant culture methods from the runtime system for *RealToStr* and *StrToReal*. If you need the localized methods you may directly access the methods in the runtime system library "`RTS.dll`".
- ⋆ The *gpcp* scanner also now uses the invariant culture methods for real literals uniformly in all cases.
- ⋆ The symbol file for the *ProgArgs* library now reveals the previously undocumented method *GetEnvVar* which (only in the *.NET* version) returns environment variable strings.

The changes to overload resolution do not constitute a breaking change, since all previously working cases will still work. However, a useful set of extra cases are handled. See also the comments in the new example program *Params.cp* in the *NETexamples* directory.

Changes from 1.3.8

The following corrections are included in the 1.3.9 release.

- ⋆ *PeToCps* extracts public key tokens from *PE*-files using new methods of *PERWAPI*. This avoids an issue with compact framework libraries.
- ⋆ *BOX* once again works correctly on *.NET* framework structs.
- ⋆ Constructors with arguments for *Component Pascal* types that extend foreign classes now work as documented.

Changes from 1.3.6

The following changes and corrections are included in the 1.3.8 release.

- ⋆ *PeToCps* has been extended to correctly deal with foreign *PE*-files from the compact framework.
- ⋆ Limited records may be extended, but only in the defining module. New error messages are attached to the new semantic checks.
- ⋆ New switch */quiet* makes *gpcp* run silently whenever possible.

- ⋆ New switch */cpsym=XXX* allows the symbol file lookup path to be varied from the command line.
- ⋆ *CPMake* may be started on a module which is not a "main" module. If a non-main module is used as a starting point a warning is issued to ensure that the choice was deliberate.
- ⋆ Uninitialized local variables of pointer type now attract only a warning.
- ⋆ Empty *CASE* and *WITH* statements no longer cause the compiler to trap, but attract a warning in the absence of an *ELSE* branch.
- ⋆ *Browse* now emits import statements in v1.3.6 extended syntax.
- ⋆ The new import syntax is disallowed when */strict* is in force.

Changes from 1.3.4

The following changes and corrections are included in the 1.3.6 release.

- ⋆ The import declaration syntax is extended to allow foreign imports to be declared using their *.NET* syntax rather than by using the canonicalized names generated by *PeToCps*.
- ⋆ Latin-8 characters are permitted in identifiers and strings.
- ⋆ Much improved error reporting based on text-spans rather than (line, column) pairs. This feature also upgrades the stepping behavior in the *GuiDebug* debugger.
- ⋆ New `/perwapi` option forces use of *PERWAPI* even when producing debuggable *PE*-files. This depends on the new version of *PERWAPI*, which can read and write `*.pdb` files.
- ⋆ A bug in the parsing of numeric tokens ending in H and L is fixed.
- ⋆ New errors are reported for numbers too large for H format, and for numbers even too large for L format.
- ⋆ A bug in the *BITS* function on integers larger than max-int has been fixed.

Changes from 1.3.3

The following changes and corrections are included in the 1.3.4 release.

- ⋆ A more flexible canonicalization of assembly names has been introduced, to allow access to assemblies with filenames containing characters illegal in *Component Pascal* identifiers
- ⋆ Fixed some incorrect cases of coercion of character arrays to native strings
- ⋆ Fixed some incorrect cases of usage for *MIN, MAX* and *INC* for short integral types
- ⋆ Fixed an error in some usages of *arrays* of procedure types

Changes from 1.3.0

The following changes and corrections are included in the 1.3.1 release.

- ⋆ A new symbol file generator *PeToCps* replaces *N2CPS*. As a result, static methods, fields and constants are available for the system value types that map into the built-in types of *Component Pascal*.
- ⋆ *Browse* displays the names of formal parameters if these are available in the symbol file. Browse has a new "*/hex*" option so as to output integer literals in hexadecimal notation. Browse has a new "*/sort*" option so as to output types and static features in sorted order.
- ⋆ *LEN* now allows an argument that is an array typename, as well as the traditional case of a variable designator.
- ⋆ New Built-in constants *INF, NEGINF* have been implemented. These may be used either as REAL or SHORTREAL values.
- ⋆ The treatment of foreign modules that overload member names with fields as well as methods are now correctly handled. This is permissible behaviour in *Java*, but not *C#*.
- ⋆ Calls of *NEW* on open arrays with multiple dimensions now correctly handle arbitrary expressions in the length arguments.
- ⋆ Extremely long method signature strings in the *JVM* emitter now no longer cause a compiler panic.

Changes from 1.2.0

The following changes and corrections are included in the 1.2.x release.

- ⋆ Support for boxing and unboxing of *CLS* value types is included.
- ⋆ The vector types have been included.
- ⋆ The parser now allows return types and formal parameters to be anonymous constructed types. The compiler gives a warning when the type so defined will be inaccessible and hence useless.
- ⋆ A string library *StringLib* has been included.
- ⋆ Some corrections have been made to the *RealStr* library.
- ⋆ The "`WinMain`" pseudo-module introduced to mark base modules for windows executables that do not start a console when launched.
- ⋆ Unsafe facilities in module "`SYSTEM`" introduced.
- ⋆ Enhanced compatability between native strings, string literals and character literals.
- ⋆ Correction to the semantics of subset inclusion tests, both versions.

Changes from 1.1.6

The following changes and corrections are included in the 1.2.0 release.

⋆ The semantics of "super-calls" were incorrect in the case that the immediate super-type did not define the method being overridden. In version 1.2 the notation "`Foo^()`" denotes the overridden method no matter how distant it is in the inheritance hierarchy.

⋆ New options have been implemented for output directories.

⋆ The default behavior for the "`/nodebug`" option is to use the direct *PE*-file writer. This is significantly faster than going through `ilasm`. Unfortunately, this new file-writer does not produce debug symbols at this stage. There is separate documentation for the *PERWAPI* component included with this release.

⋆ The permitted semantics for constructors with arguments is significantly enhanced. This is of some importance when deriving from types that do not have public no-arg constructors.

Changes from 1.1.4

The following changes and corrections are included in the 1.1.6 release.

⋆ Uplevel addressing of reference parameters is now permitted in the *.NET* release, although this has inexact semantics in some cases.

⋆ A number of corrections to the *JVM* code-emitter have been added.

⋆ The new built-in function *BOX* has been added.

⋆ Trapping of types that attempt to indirectly include themselves is improved.

⋆ An automatic renaming scheme is implemented for modules that attempt to export types with the same name as the module on the *.NET* platform.

Changes from 1.1.3

The following changes and corrections are included in the 1.1.4 release.

⋆ The copyright notice has been revised. *gpcp* is still open source, but now has a "FreeBSD-like" licence agreement.

⋆ A correction to the *Java* class-file emitter now puts correct visibility markers on package-public members. Appletviewer didn't care, but most browsers objected!

⋆ It is now permitted to export type-bound procedures of non-exported types, provided the procedure overrides an exported method of a super-type.

⋆ More line-markers are emitted to *IL* in *.NET*. This makes it possible to place a breakpoint on the predicate of a conditional statement, and have the debugger stop on the predicate rather than the next executable statement.

⋆ The type-resolution code of "`SymFileRW.cp`" has been radically revised. It is believed that the code is now immune to certain problems caused by importing foreign libraries with circular dependencies.