記号処理 32－5－1
（1985. 6. 18）

Common Lisp へのいくつかの話題

井田昌之
（青山学院大学）

Common Lisp［1］は多数の人の協力の上に生まれている。初期の論文［2］においても多数の名が記されている。Guy Steele はこれを「協調の精神に基づいて，すべての人の利用に最良と思われるLispの設計を助けるために，無欲のうちに時間を費したボランティアによってすべて開発された」という表現で［1］の日本語版への序の冒頭で記している。Common Lispの仕様を技術的に評価する場合にでもこのことの意味は大きい。ここではCommon Lispのフォーラムをすすめる上でテーマになりうると思われる点および関連資料をまとめる。動向などについては，［3］［4］［5］［6］などに記されているのでここでは省略する。

Common Lispのゴール　一口に言ってCommon Lispは，豊富な機能・表現法とデータタイプを持つ，共通性の高い言語仕様をゴールとするLisp方言である。［1］に記されたゴールは次のようなものである。①共通性（そもそもMac Lispの後継処理系，Zetalisp，Spice Lisp，NIL，S1 Lisp などへの共通仕様に始まる），②可搬性（機種独立性の高い仕様），③同一性（コンパイラとインタプリタのセマンティクスの統一），④表現能力（従来のものから，使い易い構文要素を豊富によりすぐった），⑤互換性（Zetalisp，MacLisp，Interlisp順），⑥効率（実行速度・最適化コンパイラへの配慮），⑦能力（豊富な機能，しかしツールはYellow Pagesに），⑧安定性（仕様は充分な検討を経てのちにのみ改訂）

ランダムトピックス

①コルーチン・マルチプロセスへの対応は仕様外としている。stack group 関連は除かれている。［1］は3章でこの点についてスコープ／エクステント問題とshallowとdeepのセマンティクスの点から同一性の為にはずされていることを述べている。

②クロージャの実現方法・内部表現形式は定義されていない。#'(lambda…)がクロージャを形成する。現在のZetalispやGC Lisp 1.0では(closure<変数リスト>…)として，とじる変数を利用者が指定している。また，Common Lispでは完全な閉じ込めが要求されている。その意味でactorやオブジェクト志向でいうオブジェクトと通ずる。単純に考えると大きなオーバーヘッドが予想されるが，たとえばS1のコンパイラでは束縛の解析をして，stackわりつけで済むものとheapわりつけをしなければならないものを分け，実行時の負担を軽減させている［7］。（ついでにS1では［7］によればdeep bindが採用されている。）

③変数束縛の手法は任意である。shallow／deepのいずれでもよい。むしろコンパイラの場合，いずれにせよスタック上などに別にわりつけるケースが多く，これが静的スコープルールの理由づけの基盤になっている。shallow/deep環境との整合はspecial宣言による。special変数参照はスタック上へ特殊マーカを積む手法などがあるようだ。

④多数の機構・構文・構文要素がある。少なくとも筆者らの近辺ではあまりとりあげられなかったものが多かった。（必ずしも，歴史的にはCommon Lisp初出ではない。）パッケージ，structure，リーダマクロ，多数の制御構造，ラムダリスト

キーワード…　（スタンダードLisp／Reduceという比較的小さな仕様になじんでいたので，という個人的な言い訳がある。）

⑤豊富なデータ型とそれらの階層。それらの処理機構。分数，複素数，そして4タイプの浮動小数点数。列による統一的扱い。（単純・一般）ベクタ／配列，文字型の独立，ハッシュ表，…。[8]に階層図をつくり，Guy Steeleにatest してもらったがやはりかなり複雑である。

終りに

Guy Steeleにコメントを求め，その返事が手許にあるので付録として次頁よりそれを付す。参考になれば幸いである。（ただし85年1月の時点でのものである。）

Common Lisp処理系を開発するには2つの方法があると思っている。1つは既存のCommon Lisp処理系に相談することである。これは交渉の問題であり，その根底には国際的センスが必要である。もう1つは独自開発である。このためには従来の自前のLispからの発展方法，いいかえれば立ち上がり方について検討しながら設計を進める必要がある。（全くの無からといっても何らかの下地があるはずである。）そのプロセスの一つのひな型としてGCLispの立ち上がりがあると筆者は思っている[9]。

1985年5月より電子協（JEIDA）LISP動向専門委員会が始まった。そこではCommon Lispを中心に検討が進められている。これはもちろん学術的な新規性を追求する委員会ではない。むしろ収束的な活動と理解している。標準化，あるいは共通仕様の理解とはそういうものであると思って委員長をお引き受けした。この委員会では，極力米国と連絡をとりながら進めたいと思っている。ken kahnを中心とするオブジェクト志向作業グループ（原語はsubcommitteeであるが，インフォーマルでありまたオープンメンバであるので「作業グループ」としている）の3月までのArpanet上での討議の交信録はkahnより受領し，また再配布の許可を得たのでJEIDAの希望する専門委員5名に配布した。氏名は各位の了承を得ていないのでここでは記さない。また，初期メンバリスト[10]及び交信の概要をまとめたもの[11]を配布している。私信によればkahnは日本との意見交換を求めている。そしてそれはもちろん重要なことである。残念なことに筆者にはオブジェクト志向の話まで扱う知識と余裕が今の所ない。パソコン及びサブセティングについてはGuy Steeleからもまた，Fahlmanからもこうした所は日本が得意ではないか？というコメントが届いている。（オブジェクト志向の資料の配布などについてはNTTの竹内氏が協力してくれると奥乃氏から間接的に伺っている。）

参考文献

[1] Guy L Steele Jr.他著.：Common Lisp(井田訳)；共立出版，1985年7月出版予定

[2] Guy L Steele Jr.他著：An Overview of Common Lisp；P98～107 '82 Lispコンファレンス．ACM

[3] 井田：Lispコンファレンスに出席して；bit（共立出版）PP99～101 vol.16 No.11 1984.10

[4] 奥乃：1984 Lispおよび関数プログラミング国際シンポジウム報告；IPSJ WGSYM No.29-11 P78～79 84.10

[5] 湯浅，萩谷：Common Lisp入門(1)～(3)；bit vol.17 No.4～6 1985.4～6月

[6] 井田：Common Lispは標準規格か？；日経バイト p.201～206 1985.6

[7] R.A.Brooks, R.P.Gabriel, Guy L.Steele Jr.；An optimizing Compiler for lexicaly Scoped Lisp. SIGPLAN Vol.17 No.6 PP261-275 1982.6

[8] 井田：「ポータブルバックエンド型Lispプロセッサ ALPS/IIIの構想」の図1；IPSJ. WGSYM 30-2 1984.12

[9] －：GCLISPへの検討からみたCommon Lispの要点．；JEIDA LC-2-2 1985.6

[10] －：Common Lisp object oriented subcommitteeのメンバ；JEIDA LC-1-2 1985.5

[11] －：Common Lisp object oriented subcommittee活動のあらまし；JEIDA LC-1-3 1985.5

〔付録1〕 Responses to Comments of Dr. Masayuki Ida
dated January 14, 1985 (付録2にその要訳)
on the *Common Lisp Reference Manual*
by Guy L. Steele Jr.

Your changes for typographical errors are of course all correct. Thank you for noting these.

Here are responses to specific questions. In these responses I try to distinguish *interpretation*, which is my opinion of what the book now actually says; *intention*, which is what I understood to be the intent of the committee when I wrote the text, and what I tried to convey in writing (but may have failed); and *opinion*, which is what I now think the language ought to look like (but which others have not yet agreed to).

1. *Interpretation.* To support a data type such as `(complex string)` in a COMMON LISP implementation is clearly permissible as an extension. As the text stands, I would say that if an implementation supports a data typethen the specification on page 35 would require that type to be a subtype of `common`.

   Intention. From the start I had in mind allowing only non-complex numbers as components of complex numbers (cf. page 19). The case of `(complex string)` certainly has not been considered by the committee up to this time.

   Opinion. I am unsure whether it is wise then to regard that type as a subtype of `common`. I intend to recommend to the COMMON LISP committee that the text be revised to "`(complex` x`)` where x is a subtype of `(and common number (not complex))`" or words to that effect.

2. *Interpretation.* Indeed, the book now says nothing about inclusion and disjointness relationships among `function`, `compiled-function`, and other types.

   Intention. The committee was wary of overconstraining the `function` data type lest they eliminate some unforeseen but important implementation strategy.

   Opinion. I find it very curious that `function` is not a subtype of `common`! That it is not is surely my error. I would also want arrays of functions to be a subtype of `common`. I intend to recommend to the COMMON LISP committee that `compiled-function` be required to be a (not necessarily proper) subtype of `function`, and that `function` be a subtype of `common`.

3. *Interpretation.* The book does not mention the type `compiled-function` (not, for that matter, the data type `function`) in section 25.1. It does refer more than once to a "compiled functon object." This may be a source of confusion.

   Intention. The intent is that, under suitable circumstances, the function `compile` produces an object of type `compiled-function` and returns it or uses it as a global function definition. It is also intende that the result of loading a file produced by `compile-file` is, among other things, the installation of objects of type `compiled-function` as global function definitions for functions defined at the top level of the source file.

   Opinion. I intend to recommend to the COMMON LISP committee that section 25.1 be modified to clarify the intentions listed above, and also to state explicitly that the file produced by `compile-file` is intended to be loaded sing the function `load`.

4. *Interpretation.* The diagram of the COMMON LISP type hierarchy in your paper seems to be complete and in agreement with section 2.15.

   Opinion. Your diagram appears to be a very useful way to view the COMMON LISP type hierarchy. Would you give your permission to reproduce some version of that table in a future edition of the *Common Lisp Reference Manual*?

5. There was a meeting in Monterey, California, of representatives from many institutions interested in COMMON LISP. At this meeting very few firm decisions were made, but the meeting was very important in setting general directions by which COMMON LISP would progress.

   On the matter of trademarks, it was agreed that it was probably too late to try to trademark the name "COMMON LISP" itself. I believe that DARPA intends to trademark a phrase such as "Validated COMMON LISP" and permit use of such a trademark only in conjunction with implementations that have been demonstrated to pass a series of software tests. I do not know what action has been taken on this.

6. Another outcome of the meeting was that certain subcommittees were set up, composed of volunteers, with discussions to take place via the ARPANET. Most of them are technical subcommittees charged with exploring such topics as possible official subsets of COMMON LISP, windows, graphics (bit-map and otherwise), iteration constructs (siuch as `loop`), flavors, and so on. One subcommittee, however, is responsible for proposing an administrative organization under which the development of COMMON LISP may proceed in a beneficial manner. A few such organizations have been proposed, but none has yet been adopted. Until such an organization is officially adopted, it will be difficult to makes changes the the specification that will be regarded as official.

   For the moment, no official changes to the specification have been made. The specification is discussed informally, and several of us (Steele, Fahlman, Gabriel, Moon, and Weinreb in particular, but others as well)

try to respond to questions about the language in a timely fashion, cross-checking our answers with others as necessary. I have been maintaining a list of typographical errors for a second edition; for the sake of expediency, I intend to correct such obvious errors unilaterally, without taking a formal vote (but I will circulate a list of such changes in case anyone cares to comment on them). I am also keeping a list of minor but substantive issues that should be voted upon but probably can be disposed of with little debate. Major issues should be handled by the subcommittees mentioned above.

My feeling is that we should try to produce a second edition for publication in 1986.

7. It is possible to subset COMMON LISP, of course. The question is, will it be "officially recognized" as a COMMON LISP implementation? There was a great deal of debate on this at the Monterey meeting. Some felt that the purpose of COMMON LISP to permit transportability will be defated if there are subsets. (The proponents of Ada felt this way, and no subset of Ada is officially recognized. This has led to footnotes in papers of the form, "Our language would be a subset of Ada if Ada had subsets.") Others felt that it is important to recognize at least one official subset for use on microcomputers, especially for teaching purposes. There is a subcommittee to investigate this topic and make a proposal. I believe that so far DARPA has taken no position on this; what they do will be important, because they will be administering the "official trademark," if any, and to that extent have control over the question of subsetting.

8. It is a good question who will verify COMMON LISP implementations. I believe that DARPA is funding or will fund an effort to develop and maintain a validation suite of test software. Various implementors such as DEC have volunteered to contribute their test code to such a validation suite of others will do likewise. There will have to be established a facility with enough resources to perform verification of implementations.

[付録 2] Guy L. Steele Jr. への質問・要旨

1. "(complex x) where x is a subtype of common" (p35, l12) とあり, x は数型以外のもの, 例えば string でもよいのか?

2. 型としての function/compiled-function の扱いがおちているのではないか. 例えば 2.15 には書かれていない.

3. 25.1 節において コンパイルされた結果の compiled-object と compiled-function type の間の関連が記述されていない.

4. type hierarcy の図をまとめ同封した([8]図1)。これで正しいか, なにか落ちていないか.

5. Common Lisp の商標に関する議論はその後どうなったか.

6. 仕様の改訂はどうやっていつ行なわれるか. 進行中か. マルチウィンドウ/ビットマップの件はどうなるか?

7. 言語のサブセッティングは行なわれるか. kernel data type の観点ではどうか.

8. 誰が Common Lisp の異なる処理系に対して verification を行なうのか?

以上