

Emacs における高精度コード補完機能の開発

1. 背景

ユーザーが自由に拡張・カスタマイズできる高機能なテキストエディタとして普及している Emacs だが、残念なことに主な用途であるプログラミングにおいて重要な機能となるコード補完機能が非常に使いづらいという問題がある。コード補完機能の UI としてはポップアップメニューによる補完がメジャーであるが、Emacs のコード補完機能の UI はレガシーなもので、簡単に言えば、視覚的効果が全くないため次に何が補完されるのか予測できない。

また、Eclipse や Visual Studio が持つコード補完機能はいわゆる「文脈」を考慮する有意味な高精度コード補完が可能で、これによってさらに効率的に作業することが可能になる。ところが、Emacs においてはそういった「文脈」を考慮する高精度コード補完が不可能なため、コードを書いてコンパイルするといったプロセスを繰り返したり、インターネット検索などでわざわざマニュアルを調べたりしなければならない。それは時間の無駄であるし、Emacs やそのユーザーがもつ本来の高い生産性を発揮できない。

2. 目的

クリエイターの知る範囲では、Emacs ユーザーは他のプログラマに比べて特に優れた能力を持っていることが多々ある。これは単なる主観的な評価でしかないが、仮にそれが正しいとして、そのようなユーザーに高精度コード補完機能という最高の道具を与えたら、一体どれほど生産性を向上させるか想像に難くない。

また、高精度コード補完という現代的な機能を通じて Emacs をより多くの人に知ってもらい、単なる古臭いテキストエディタではないということを認知させてユーザーを増やし、Emacs が世界一のテキストエディタであれるよう、また世界一のコミュニティを持てるようにするのがもう一つの目的である。

3. 開発の内容

本プロジェクトでは、Emacs のポップアップ型コード補完機能である `auto-complete.el` の改善・拡張、Ruby のための高精度コード補完機能である `RSense`、C/C++ のための高精度コード補完機能である `GCCSense` を開発した。

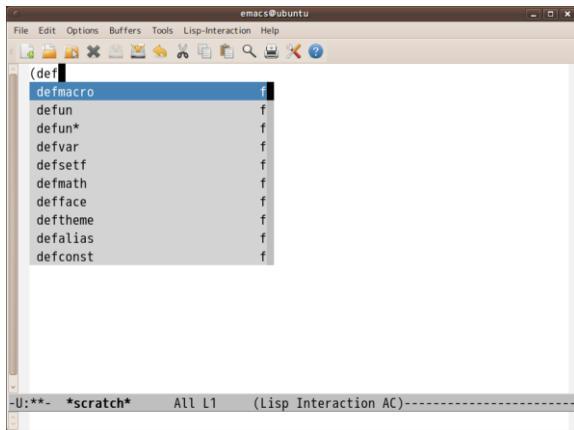


図 1. auto-complete.el ポップアップ補完

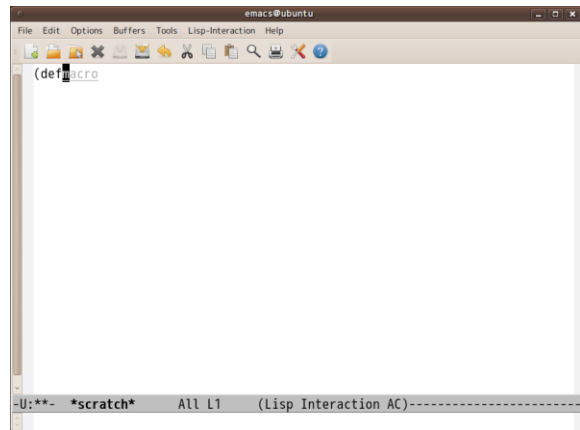


図 2. auto-complete.el インライン補完

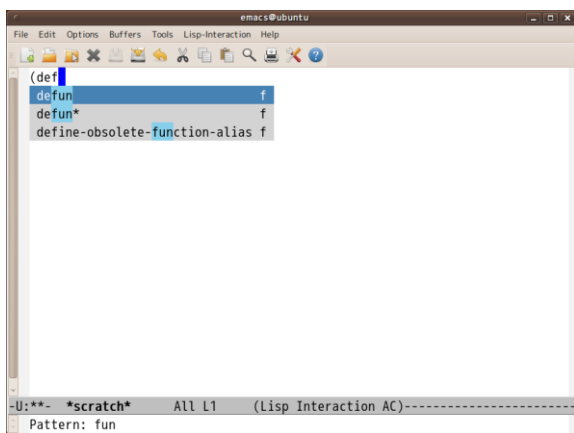


図 3. auto-complete.el インクリメンタル補完

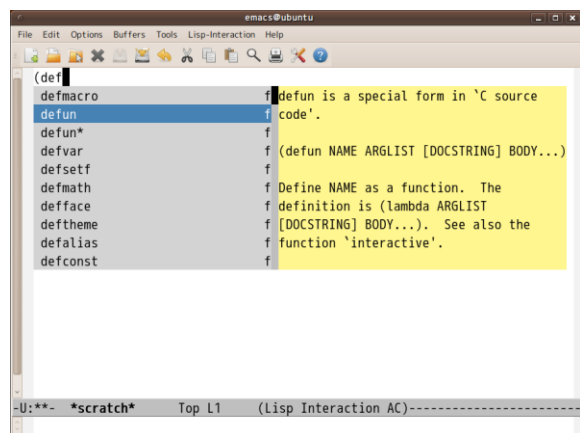


図 4. auto-complete.el ホップアップヘルプ

RSense は単独のコマンドラインプログラムとして動作し、テキストエディタから適切に呼び出すことで高精度コード補完を実現する。対応エディタは Emacs、Vim、TextMate、Redcar、Xyzzy、秀丸となっている。Java で実装されているためプラットフォームは選ばない。

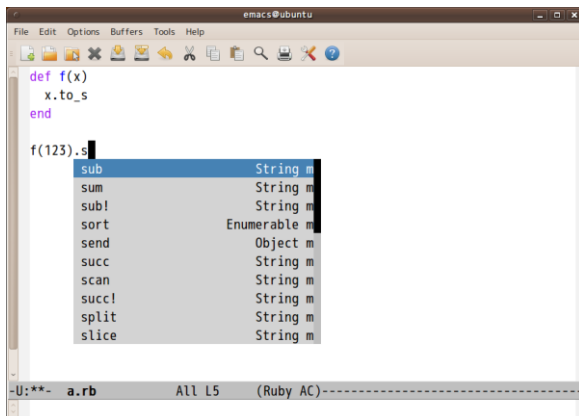


図 5. RSense 高精度コード補完

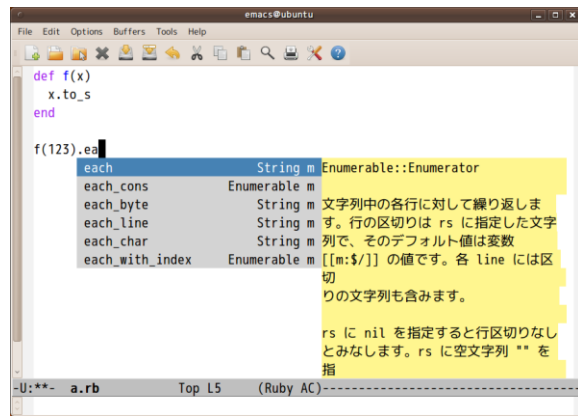


図 6. RSense 補完候補ポップアップヘルプ

GCCSense は GCC にコード補完機能を追加したカスタム GCC を利用する一連のプログラム群である。GCC のコマンドを記録・再生する gccrec、自動的にプリコンパイルヘッダーを適用する autopch が含まれる。テキストエディタはこれらのプログラムをうまく組み合わせることで、高速な高精度コード補完を実現する。対応エディタは Emacs、Vim、Redcar となっている。

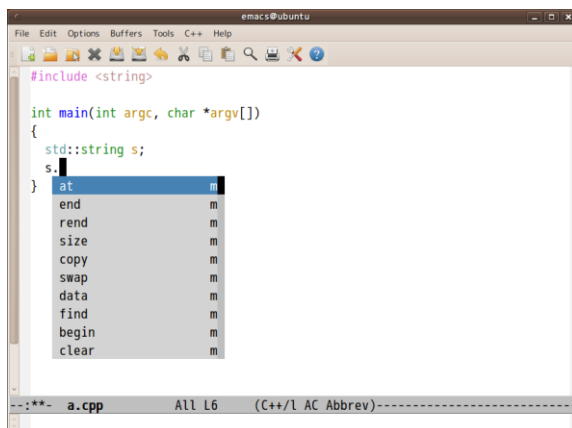


図 7. GCCSense 高精度コード補完

4. 従来の技術(または機能)との相違

auto-complete.el は従来のコード補完 UI とは比べものにならないほどの優れた UI を提供している。インライン補完とポップアップ補完を区別することにより、空気のように使える UI を実現している。同様の機能を提供する拡張として company-mode が挙げられるが、これはユーザーの作業を邪魔する傾向があり、かなり使いづらいと感じさせるものである。

RSense の競合として、rcodetools や Vim の標準機能、3rdRail や RadRails、RedMine などの製品が存在しているが、どれも非常に簡単なコードですらコード補完できないという精度上の問題がある。一方 RSense は、若干の

の制限はあるものの、2000 行程度のスクリプトでテストした範囲では、ほぼ 100%の精度を実現した。

GCCSense の競合として、Visual C++や Eclipse の CDT プラグインがある。セグメントの違いはあるが、単純に精度を比較すれば GCCSense は CDT プラグインよりかなり優れており、Visual C++と同等かそれ以上の精度になっている。また、WebKit や Firefox、Google V8 などの巨大な C++プロジェクトでも問題なくコード補完できたことから、実用の面でも問題ないと言える。極論を言えば、コード補完のバックエンドとなる(カスタム)GCC がコンパイルできるコードであれば、ほぼ 100%の精度でコード補完が可能ということになる。

5. 期待される効果

auto-complete.el や RSense、GCCSense を利用することにより、Emacs 上でのコーディング作業が効率化される。それはつまり Emacs ユーザーの生産性を向上させ、対象製品のコストの削減につながる。また、これまで Emacs を使いたくても高精度コード補完機能がないために、しかたなく Eclipse を使ってたというユーザーを再び Emacs に呼び戻すことが可能になるため、その分の生産性の向上も望めるだろう。さらに、auto-complete.el や RSense、GCCSense はプログラマーから瑣末な作業を忘れさせて、プログラミングの楽しさを演出する(楽しさは重要である)。その他、教育目的の利用も考えられる。

6. 普及(または活用)の見通し

ダウンロード数から割り出した非常におおまかな統計によると、auto-complete.el を利用している日本人は 700 人ほどいる。RSense や GCCSense はもっと少ないだろうが、徐々に広まりつつある。また、海外でも一定の注目を集めているため、海外における普及も十分可能だと考える。戦略としては、私がもつ Emacs や Vim のチャンネルを利用して、Emacs ユーザーや Vim ユーザーに普及させたあと、品質向上に従って、さらに多様なテキストエディタに対応することで普及させようと考えている。その際、Eclipse のサポートは重要になるだろう。

7. クリエータ名(所属)

松山 朋洋(専修大学 経営学部)

(参考)関連URL

未踏期間中に制作したソフトウェアの配布やマニュアルの公開を行っている。

<http://cx4a.org/>