# 2010 年度 未踏 IT 人材発掘・育成事業 採択案件評価書

## 1. 担当PM

首藤 一幸 PM
（東京工業大学 大学院情報理工学研究科 数理・計算科学専攻 准教授）

## 2. 採択者氏名

チーフクリエータ： 坂本 一憲（早稲田大学大学院）
コクリエータ： 太田 大地（早稲田大学大学院）
コクリエータ： 岩澤 宏希（早稲田大学大学院） *2011 年 3 月まで
コクリエータ： 大橋 昭（早稲田大学大学院）

## 3. 委託金支払額

1,792,000 円

## 4. テーマ名

複数言語対応のソースコード処理ツールのフレームワークと利用例

## 5. 関連Webサイト

http://www.unicoen.net/

## 6. テーマ概要

近年、ソースコードを対象とした解析や変形の処理ツールの発展がみられる。例えば、メトリクス測定ツールやアスペクト指向プログラミング（Aspect Oriented Programming、以降 AOP）処理系が挙げられる。

処理ツールは、一つのプログラミング言語（以降、言語）に特化して開発されることが多く、複数言語に対応した統一的な処理ツールが存在していない。複数言語を用いたソフトウェア開発プロジェクトが増加しているが、ユーザがこうしたプロジェクトにおいて処理ツールを利用することは困難である。

一方、処理ツールの実装はコンパイラの実装と同程度に困難である。さらに、様々な言語に対応した様々な処理ツールが別々に開発されており、莫大な開発コストがかかる上、開発者間で処理ツールに関するノウハウの共有が進んでいない。

本プロジェクトで開発する、複数言語対応の処理ツールフレームワーク、題して UNICOEN は、言語対応の実装と処理ツールの実装を分離することで、対応言語と処理ツールにおける多対多の関係を対応言語とUNICOEN／処理ツールとUNICOEN の一対多の関係に簡略化する。これにより、処理ツールの開発コストを大幅に削減して、複数の言語に統一的に対応した処理ツールの開発を支援する。さらに、UNICOEN の利用例として、既存手法と比較して容易に幅広い言語に対応可能な AOP 処理系を実装することで、UNICOEN の有用性を証明する。

本プロジェクトは、UNICOEN が対応する全ての言語に対応した統一的な処理ツールを低コストで開発できることを目的とする。これによって、開発者が利用可能な言語の選択肢を広げ、言語間の垣根を崩し、処理ツールにおけるノウハウ共有を促す。処理ツールと言語の発展から、ソフトウェア産業界全体、それに関わる全てのインフラやシステムの品質向上を図ることを目指す。

## 7. 採択理由

プログラムの解析や変形、例えば、規模や複雑さといった指標の測定や、アスペクト指向プログラミングのための変形を行うツールは、これまで、対象とするプログラミング言語ごとに開発されてきた。

このプロジェクトでは、様々な言語を統一的に扱える枠組みを開発する。

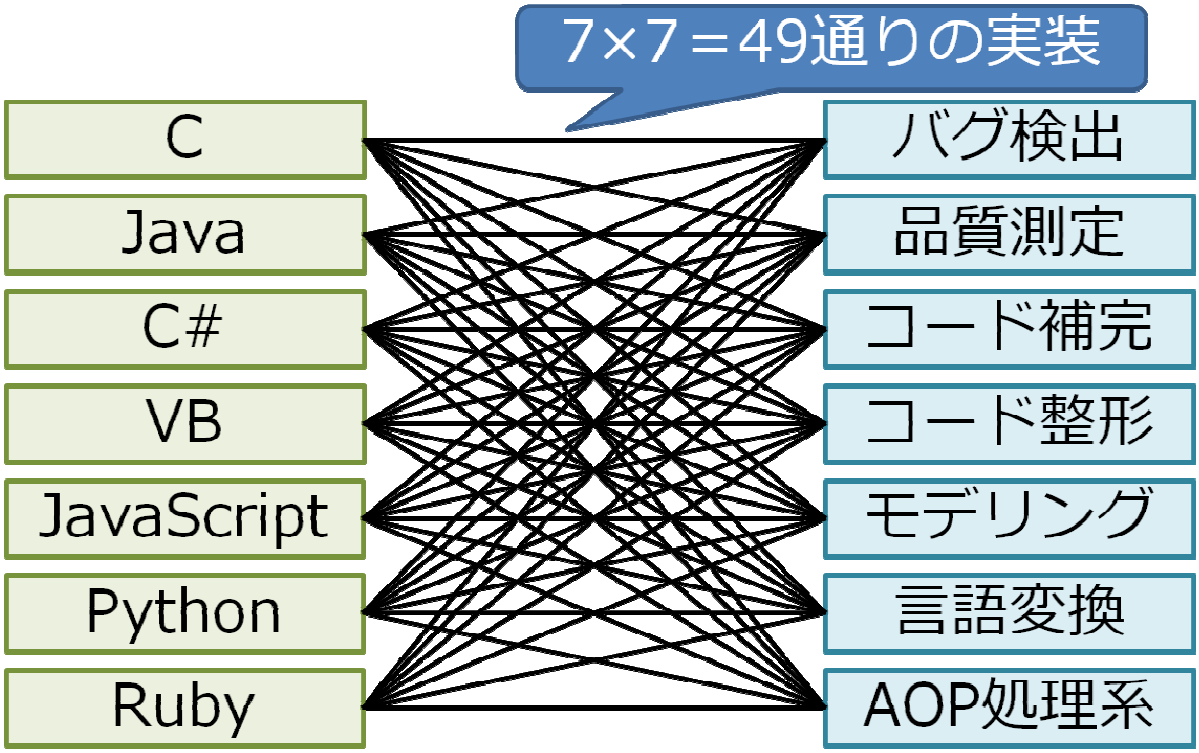
それなりの規模になる野心的な提案である。

言語や処理内容に依存する部分を小さく保ちながら、なおかつ、できることを制限せずに、使い勝手のいい枠組みとするには、設計センスが問われる。こうした設計やチーム開発においては、数字に表れにくい様々なトレードオフ、判断を経験することだろう。それらを形式知に持ち上げることをも、狙って欲しい。

ソフトウェア工学のためのソフトウェア工学にとどまらず、エンジニアを幸せにする成果を期待している。

## 8. 開発目標

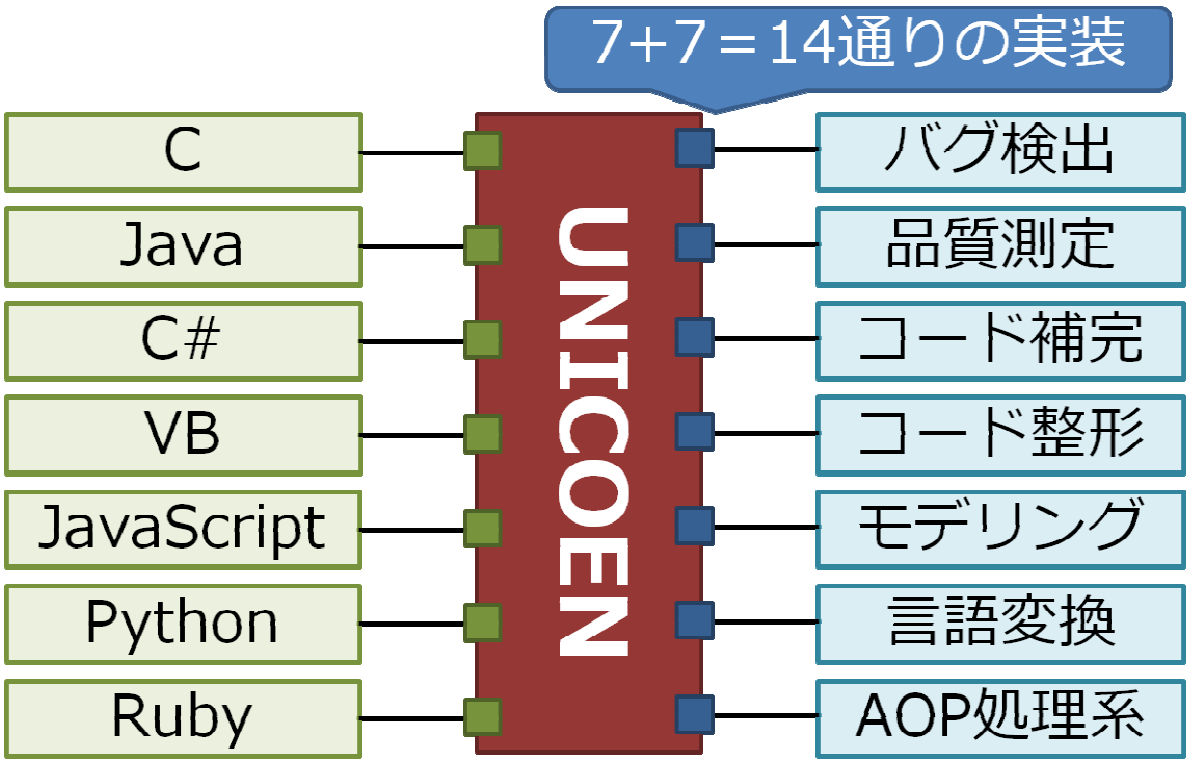
現在、数多くのプログラミング言語があり、そのそれぞれについて、多くの処理ツールが言語毎に開発されている。例えば、バグパターン検出ツールの FindBugs は Java 言語のみに、JSLint は JavaScript 言語のみに対応している。また、AOP 処理系の AspectJ は Java 言語のみに、AOJS は JavaScript 言語のみに対応している。このように、言語と処理ツールの間には多対多の関係がある。例えば、各処理ツールが１つの言語のみに対応していて、全ての処理ツールと言語の組み合わせを考える場合、下図のように「言語の種類数」×「処理ツールの種類数」通りの実装が必要になる。




図：対応言語と処理ツール間の多対多の関係

このことは、莫大な開発・保守コスト、同種のツール間の差異、といった問題を生んでいる。

本プロジェクトでは、この問題を解決するために、ソースコード処理フレームワーク UNICOEN（下図）を開発する。UNICOEN は *改善 1*：言語非依存な統合コードモデルを提供して、その上で、*改善 2*：統合コードモデル上の汎用的な共通処理をコールドスポットとして提供する。



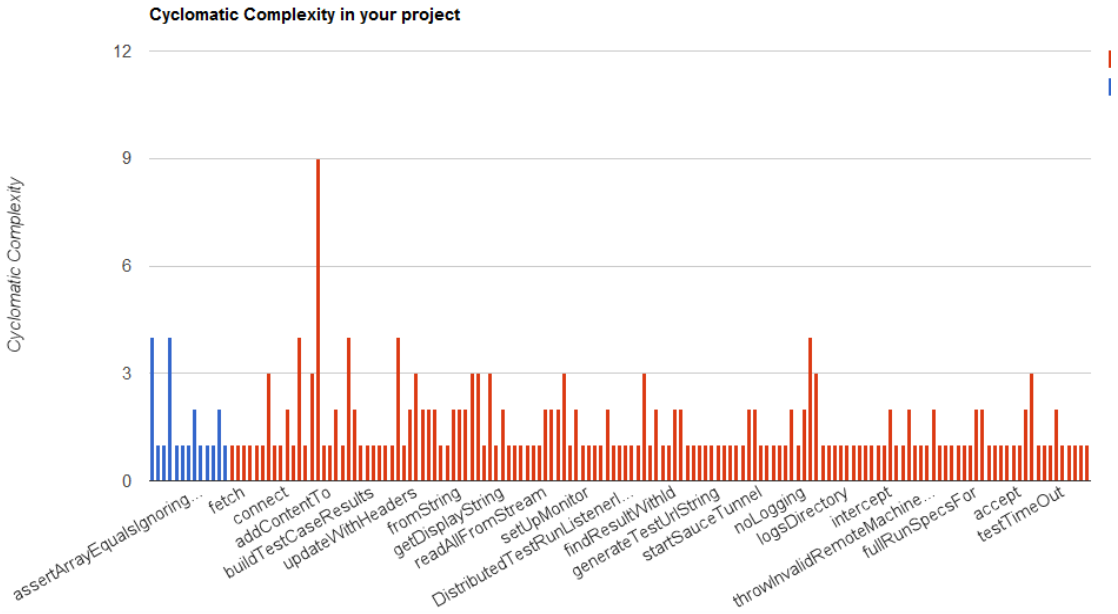
図：対応言語と UNICOEN 間、処理ツールと UNICOEN 間の多対一の関係

## 9. 進捗概要

次のソフトウェアを開発した：

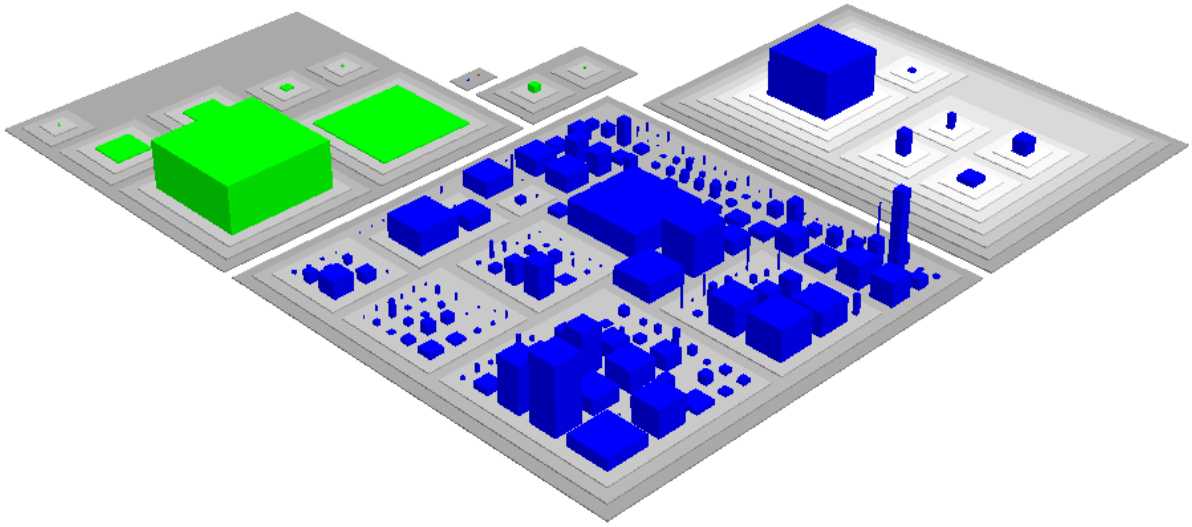
- ソースコード処理フレームワーク UNICOEN
  - 7 つのプログラミング言語に対応
- ソースコード処理ツール 3 種
  - アスペクト指向プログラミング（AOP）処理系 UniAspect
  - ソフトウェアメトリクス可視化ウェブアプリケーション UniMetrics
    複数言語で構成されるソフトウェア全体の複雑度を測定・可視化
  - ソフトウェアの構成可視化ツール CodeCity の 7 言語混在表示対応
    もともとの CodeCity は 3 言語（Java, C++, C#）に対応している。可視化対象ソフトウェアを UNICOEN を介して事前処理することで、UNICOEN が対応する 7 言語について、また、複数の言語で書かれたコードが混在するソフトウェアについても可視化できるようになった。

UniMetrics を用いて、Java と Ruby のソースコードを含むソフトウェアサイクロマチック数（条件分岐数+1）を表示した結果が、下図である。




図：UNICOEN で開発した McCabe の複雑度（サイクロマチック数）の測定ツールの実行結果

Java と JavaScript のソースコードを含むソフトウェア JsUnit を、UNICOEN で事前処理して CodeCity で可視化した結果が、下図である。



図：UNICOEN と CodeCity による Java と JavaScript 言語で開発された JsUnit の品質の可視化

## 10. プロジェクト評価

複数種類のプログラミング言語をまたがってソフトウェアを測定・処理するための基盤、というもの自体が、これまでほとんどなかった。類似ソフトウェアとして思い浮かぶのは、多言語に対応したコンパイラ（例：GCC）やコンパイラフレームワークであるが、複数の言語に共通する中間表現が低レベル（例：ループは条件分岐に変換済）であったり、高レベル中間表現を持つコンパイラフレームワークは対応言語数がとても少なかった（例：COINS は C と Fortran に対応）。コンパイルではなく、ソースコードに対する測定・処理を目的とすることで、プログラムの意味の細かいところを苦労して扱う必要がなくなったり、高レベル表現さえ用意すればよくなり、その代わりに多言語への対応をしやすくなった、ということである。

異なる言語で書かれたプログラムを同じ土俵でフェアに比較する、ということは、これまでほとんどできなかった。例えば可視化ツール CodeCity であれば、CodeCity 自体が対応する Java, C++, C#という 3 言語に限定されており、またこれら 3 言語は同じ C 言語をルーツとしているため、測定結果も似かよると予想される。それに対して、UNICOEN を用いることで UNICOEN が対応する言語であれば等しく処理できるようになった。しかもその CodeCity 用事前処理ツールはわずか 370 行で書けている。UNICOEN によって、言語をまたがったソフトウェアの比較が初めて現実的になったと言える。

## 11. 今後の課題

- 普及、利用者の獲得、そのための宣伝
- UNICOEN 自体の可能性の探求
  - 多言語で構成されているソフトウェアの測定・処理
  - 言語が異なるソフトウェア間・ソフトウェア群の公正な比較
  - 言語ごとの、ソフトウェアの性質の調査
    プログラミング言語がソフトウェアの性質に与える影響を、定量的に示す。

クリエータの中には、研究業界での業績を積むべき立場の者もいる。実際、UNICOEN はソフトウェア工学の新しい可能性を拓き得る成果であって、研究業界でも充分に認められるものと信じている。しかし一方で、プロジェクトの最終ビジョンとして掲げた「全てのソフトウェアエンジニアに笑顔を」という、実際のエンジニアリングに対して貢献しようという志を忘れないで欲しい。