京都産業大学 コンピュータ理工学部 青木研究室 第３回中間発表
２０１２年１０月２４日(水)　15:00～18:15　第２実験室棟 3階 65実験室

スパゲッティを題材としたソフトウェアプログラム可食化に関する研究

（中間発表３）

京都産業大学 コンピュータ理工学部 ネットワークメディア学科

宮崎 雅文

g0947424@cse.kyoto-su.ac.jp

目次

進捗状況

あるでん亭 新宿店「なすトマト」 ― 2012.10.02

スパゲッティを脱して

スパゲッティ調査を終えた（前回）

↓

（最後の夏休み）

↓

研究活動再開

前回からの残務

- 先行研究の調査（可食化、スパゲッティプログラム）
- プログラムの性質の調査
- 指標の検討
- 解析部の実例の作成

本来すべきこと

- 「可食化」の
  完成（レシピ生成の実現）の
  目処が立っていること

現在の状況

スパゲッティ調査を終えた（前回）

↓

（最後の夏休み）

↓

研究活動再開

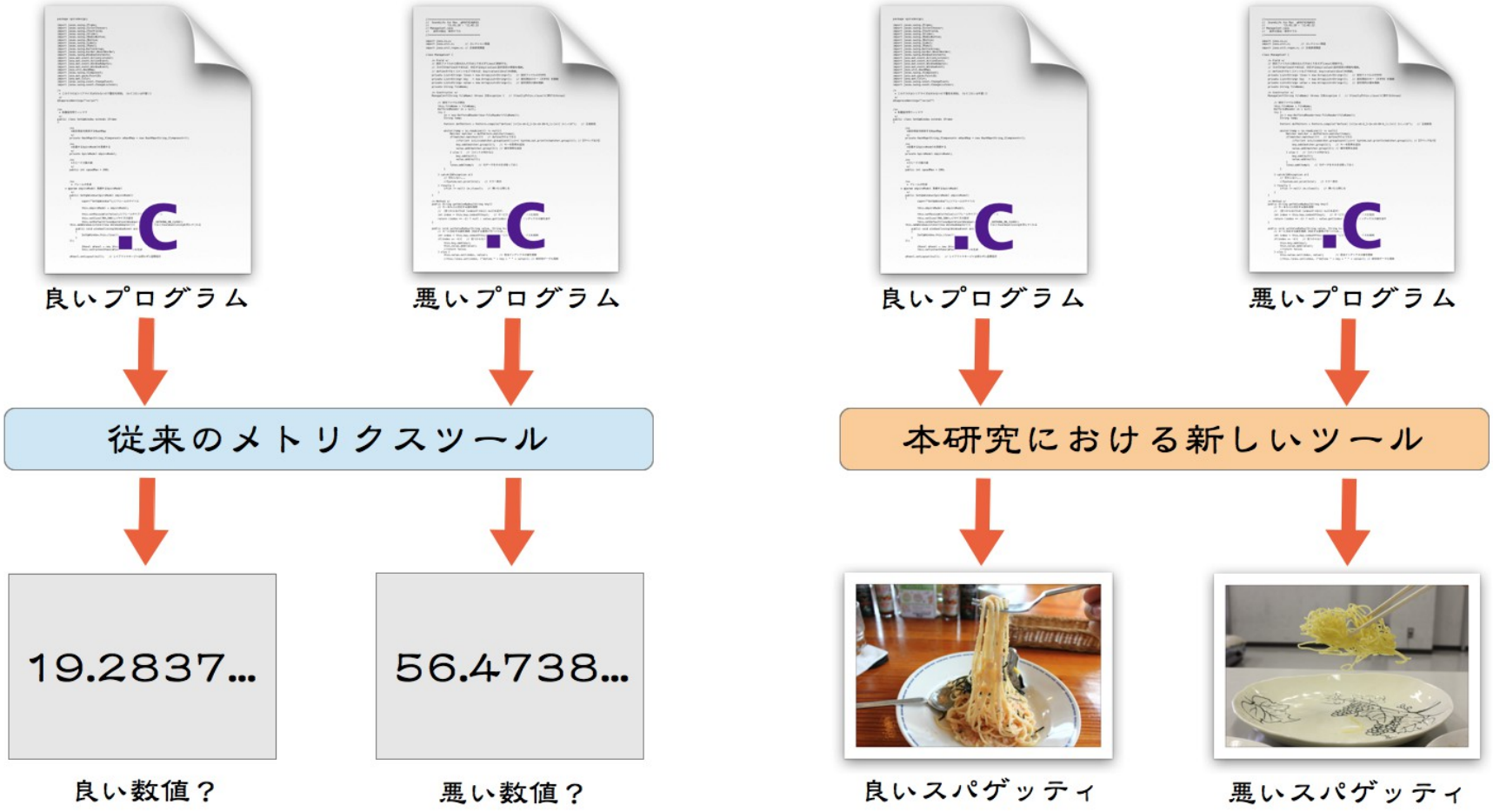
前回からの残務

- ○ 先行研究の調査（可食化、スパゲッティプログラム）
- ○ プログラムの性質の調査
- ○ 指標の検討
- × 解析部の実例の作成

本来すべきこと

- × 「可食化」の完成（レシピ生成の実現）の目処が立っていること

初見の方々へ



「可食化」という新しいアプローチの可能性を探る研究です。

プログラムの性質

プログラムの古今

【昔々】 職人技の時代

低級言語（機械語）で、実行効率の良い（gotoだらけの）ものを。

システム規模拡大で職人技も限界 → スパゲッティプログラム発生

【近世】 構造化プログラミング

エドガー・ダイクストラの基本３構造：「順次」「分岐」「反復」

構築技法が整備されてきて、さらにシステム規模は膨張。。。

【現代】 銀の弾丸を求めて

Javaだ、VBだ、ウェブアプリだ、オブジェクト指向だ、LLだ...

決定的な打開策を探し求め、どこへやら...

プログラムの良し悪し

雑誌に見られる「良いプログラム」という言葉

- 実行効率の良いプログラム
- 移植性（ポータビリティ）の良いプログラム
- 見通しの良いプログラム
- 使い勝手の良いプログラム
- 反応の良いプログラム
- 依存性が少なく良いプログラム
- 気持ちの良いプログラム
- 可読性の良いプログラム

「良い」の側面は様々！

指標 （プログラムの性質）

指標名	説明
LOC	Lines of Code。コードの行数。SLOC（Source LOC）とも。計測の規則も様々で、以下にはその例を幾つか掲載する。関数毎・クラス毎・パッケージ毎で計測する。
Effective LOC	eLOCとも。空行・コメント行・括弧だけの行を除いてカウントした数値。
Pyhsical LOC	物理LOC。ソースファイルに記載されいているプログラムの行数をそのまま計測した数値。
Logical LOC	論理LOC。lLOCとも。プログラムにおける文（statement）の数を表す数値。1行に複数文書かれている場合も文の数だけカウントする。
Comments Lines	コメントの行数。
Blank Lines	空行の行数。
FP (LOC由来)	LOCの値から算出したFP（Function Point：ファンクションポイント）値。機能数とその複雑さによって算出した数値。
Number of Input Parameters	引数の数。
Number of Return Points	関数が戻る場所（return）の数。
Interface Complexity	Input Parameters + Return Points。インタフェースの複雑さ。
Cyclomatic Complexity	循環的複雑度。線形独立な経路の数。循環の数 + 1。
{Functional, Class, Namespace/Package} Complexity	Interface + Cyclomatic。
Average, Maximum, Minimum	関数ごとのLOCについて算出する平均値・最大値・最小値。
Number of Data attributes	属性（フィールド）の数。アクセス修飾子（public, private, protected）毎に数え分ける。
Number of Methods	操作（メソッド）の数。アクセス修飾子（public, private, protected）毎に数え分ける。
Number of Functions	関数（メソッド？）の数。クラス毎・パッケージ毎で計測する。
Depth of Inheritance Tree	継承の階層数。
Number of Classes	クラス数。
Number of Base Classes	基底クラス（親クラス）の数。
Number of Derived classes	派生クラス（子クラス）の数。
Derived/Base Class Ratio	派生クラス数と基底クラス数の比率。派生クラス数÷基底クラス数で求める。
{Maximun, Average} Inheritance Depth	継承の階層数の最大値・平均値。

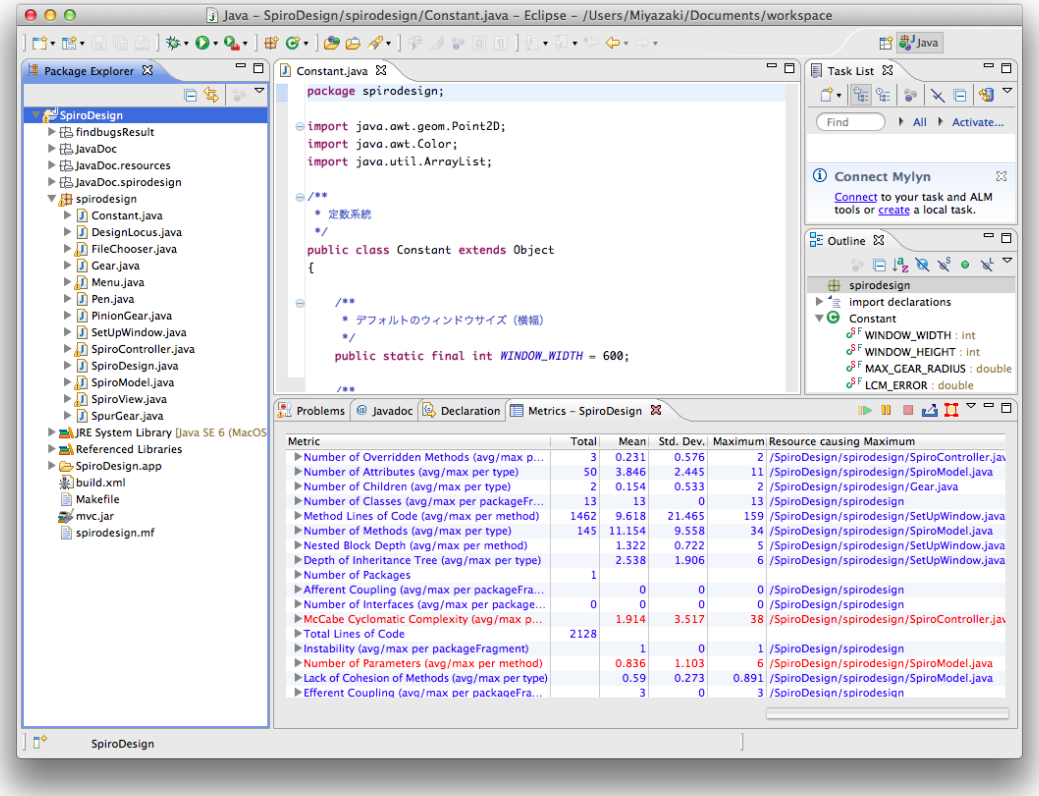
指標名	説明
Lines	空行を除いた行数を表す数値。
Statements	セミコロンで終わる文と、クラスやメソッドの定義文の合計を表す数値。
% Branches	おそらく、Statements中に存在するif文の割合を表す数値。
Calls	メソッド呼び出しの回数を表す数値。
% Comments	コメントが占める割合。
Classes	プログラム中のクラス定義数（内部クラス数を含む）を表す数値。
Methods/Class	メソッド数とクラス数の比率。メソッド数÷クラス数で算出する。
Avg Stmts/Method	1メソッドあたりの平均Statements数。
Max Complexity	おそらく、各メソッドのCyclomatic Complexityのうち、その最大のものを表す数値。
Max Depth	おそらく、{}（ブロック）の最大ネスト数。

指標名	説明
NORM	Number of Overridden Methods。オーバーライドしているメソッドの数。
NOF	Number of Attributes (Field)。属性（フィールド）数。
NSC	Number of Children。子クラス数。
NOC	Number of Classes。総クラス数。
MLOC	Method Lines of Code。メソッドのLOC。
NOM	Number of Methods。メソッド数。
NBD	Nested Block Depth。ブロックのネスト数。
DIT	Depth of Inheritance Tree。継承の階層数。
NOP	Number of Packages。パッケージ数。
CA	Afferent Coupling。依存しているパッケージ外のクラス数。
NOI	Number of Interfaces。インタフェースの数。
VG	MaCabe Cyclomatic Complexity。循環的複雑度。
TLOC	Total Lines of Code。コードの行数。
RMI	Instability。（パッケージの）不安定性。Ce / (Ca + Ce)で算出する。
PAR	Number of Parameters。引数の数。
LCOM	Lack of Cohesion of Methods。あるクラスの凝集性の欠如を表す。小さいほど凝集度は大きく、メソッドの強度が高いことを表す。
CE	Efferent Coupling。依存しているパッケージ内のクラス数。
NSM	Number of Static Methods。スタティックメソッド（クラスメソッド）の数。
RMD	Normalized Distance。
RMA	Abstractness。パッケージの抽象度。全クラス及びインタフェース中の抽象クラス及び抽象インタフェースの割合。
SIX	Specialization Index。クラスの特殊化指標の平均値。NORM * DIT / NOMで算出する。
WMC	Weighted methods per Class。あるクラスに定義されているメソッドのCyclomatic Complexityの総和。大きいほど複雑であり、メンテナンスのコストがかかることを示唆する。
NSF	Number of Static Attributes。スタティックフィールド（クラスフィールド）の数。

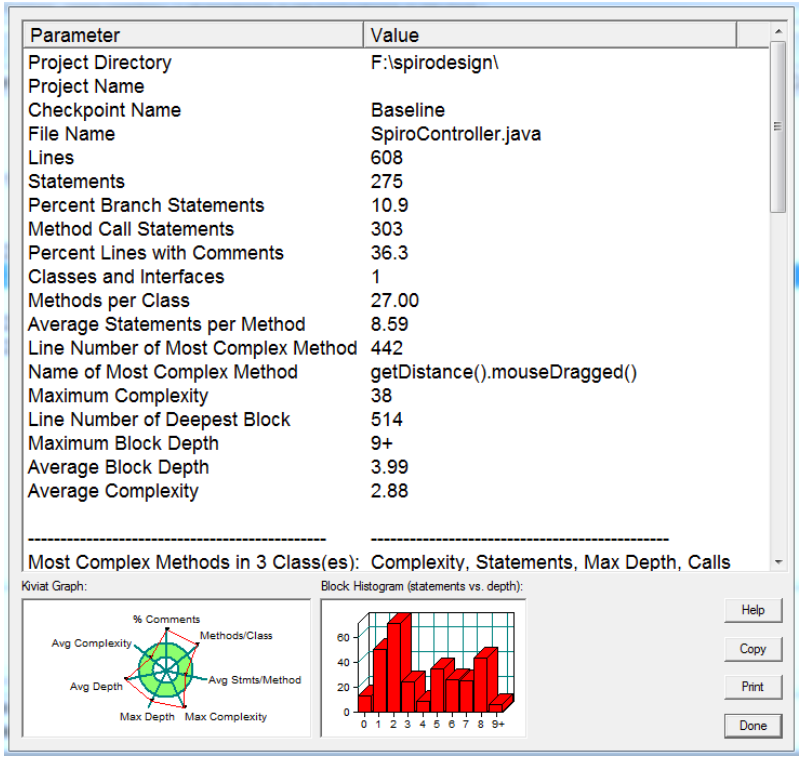
プログラムの計測と写像

既存のメトリクスツール

Eclipse Metrics Plugin



SourceMonitor



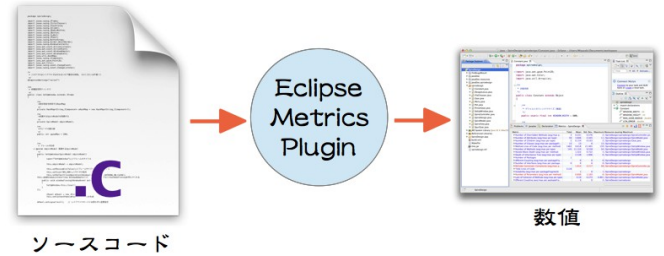
基本的に文字ベース。良くてもグラフ表現。非直感的。存在感もない。

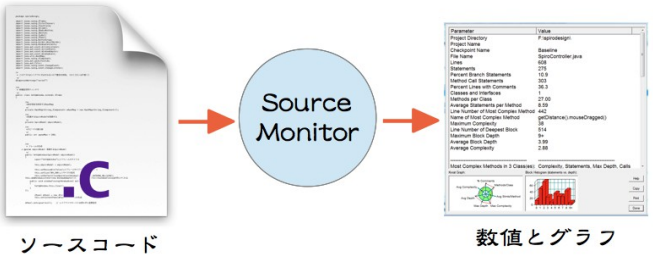
No Silver Bullet

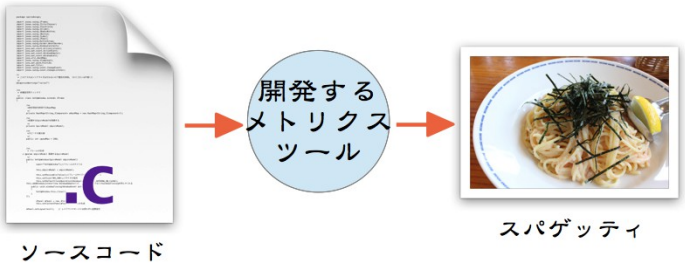
「餅は餅屋」

使う指標は使う人が決めるべし

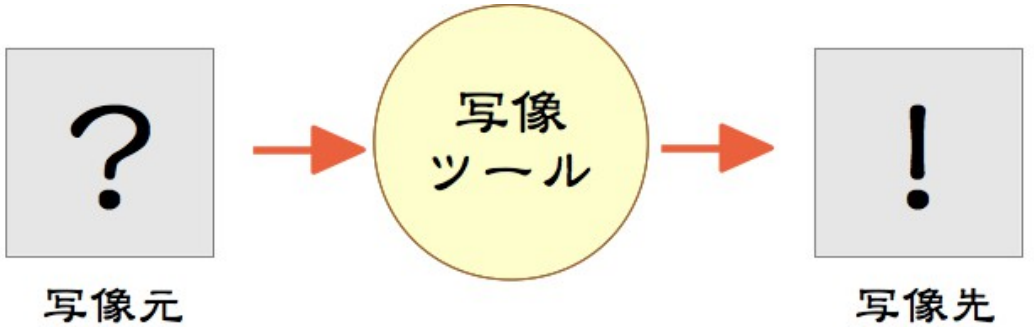
系から系への変換：写像ツール







共通の構造を抜き出してみた



プログラム → ラーメン

ウェブサイト → スパゲッティ

論文 → 分子間運動

スパゲッティへの写像

プログラミング初学者向けのメトリクス

寺子屋で垣間見た「初学者のプログラム」を活かして計測する。

「初学者にしては良いプログラム」or「初学者らしい悪いプログラム」

プログラムの指標値	スパゲッティの指標値	関係
関数・メソッド宣言数/行数	作り置き時間	負の相関
変数・関数の名称	スパゲッティ料理の名称	正の相関
インデント・スペース	ゆで時間・太さ	正の相関
コメント記述量	劣化しやすさ	負の相関

関数・メソッド宣言数/行数：作り置き時間

良いプログラム ： 良いスパゲッティ

機能ごとに
関数分けされた
Cプログラム

悪いプログラム ： 悪いスパゲッティ

main関数のみの
Cプログラム

変数・関数の名称：スパゲッティ料理の名称

良いプログラム ： 良いスパゲッティ

currentNode → ミートソース スパゲッティ

悪いプログラム ： 悪いスパゲッティ

cn → スパA

selectedCurrent NodeSelectedByUser OnPrimaryWindow → 賀茂なすと香草のスパゲティ 京都北山風サルサ・ディ・ポモドーロを添えて

インデント・スペース：ゆで時間・太さ

良いプログラム ： 良いスパゲッティ

インデントが
きれいに整った
プログラム

太さが揃っている

悪いプログラム ： 悪いスパゲッティ

インデントを
蔑ろにしている
プログラム

太さがバラバラ

コメント記述量：劣化しにくさ

良いプログラム ： 良いスパゲッティ

コメントのあるプログラム

多少の作り置きが可

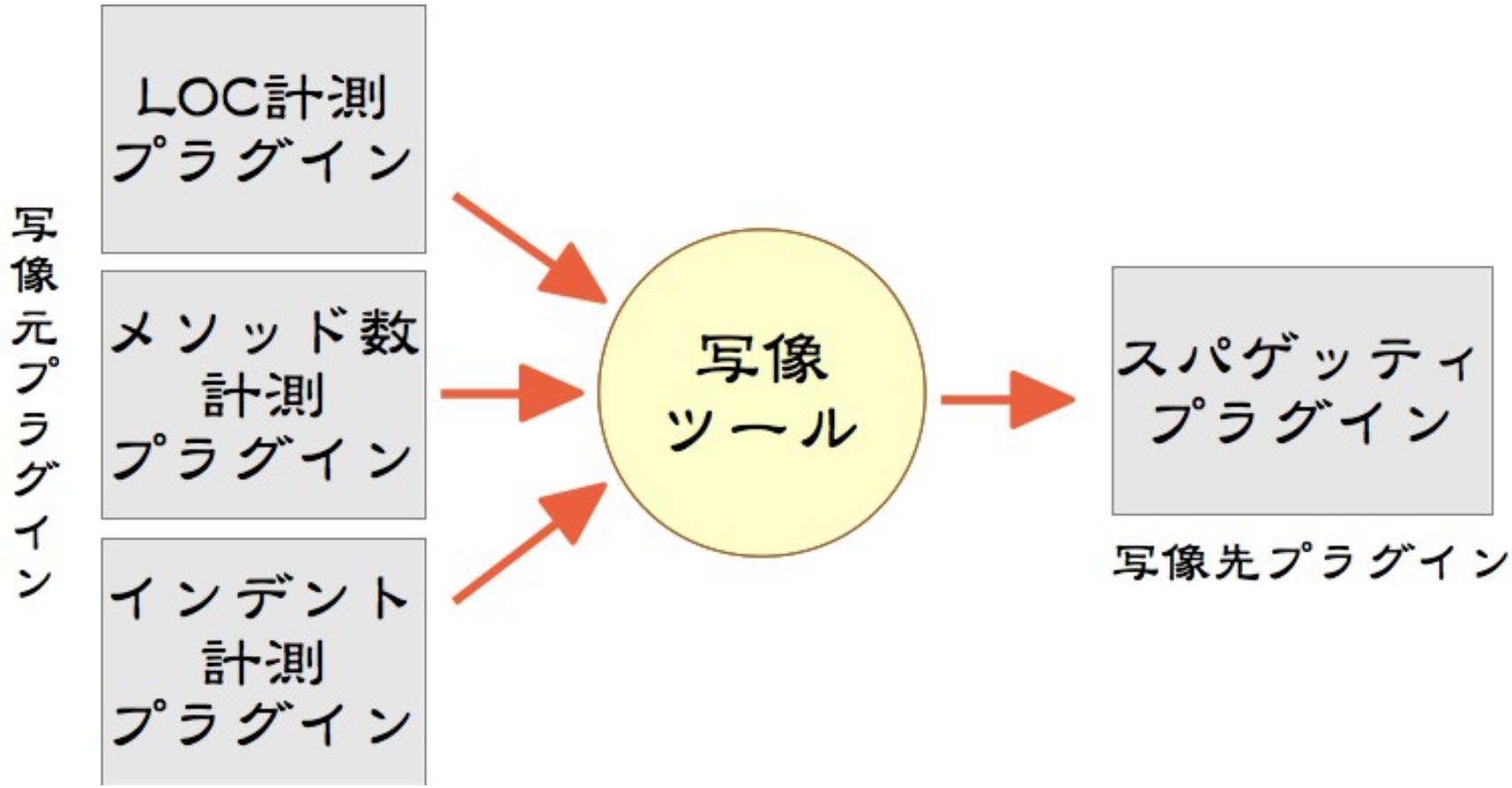
悪いプログラム ： 悪いスパゲッティ

コメントのないプログラム

すぐに乾燥(劣化)する

新しいメトリクスツール

実装に至るまでの諸問題



位相幾何的（トポロジカル）な構造をどのように表現して、

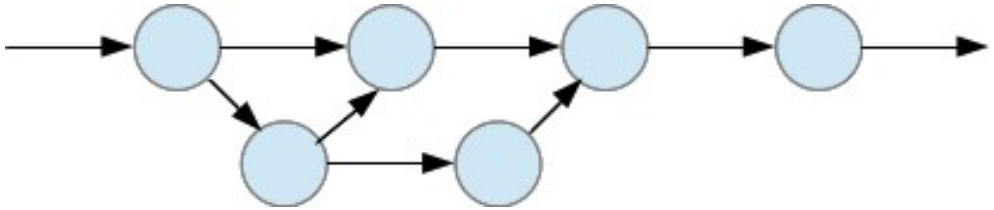
写像ツールに対してどのように伝えよう...

案１：MakeとXMLを利用する

- それぞれのプラグインはMakeに対応している。
- make xmlと叩けば、計算した指標値をXML形式で出力する。
- 写像ツールはプラグインに対してmake xmlと叩き、指標値を取得。
- make testと叩けば、人間が読める形で出力する。単体で活用可能。

案２：論理型の言語を開発する

- 「xプラグインの出力をyプラグインの入力として繋ぐ」ように表現
- forkの構造もjoinの構造も簡単に書き表せる。



- 言語処理系作ってみたい。。。

これからの開発計画

モックアップ

まずは試作品をSmalltalkで拵えるか、どうするか、迷走中。

Webアプリケーション

ウェブブラウザがあれば手軽に利用できるAnother HTML-lintに倣って。

ウェブサービスとして公開したい。これは是非とも遂げたい。

デスクトップアプリケーション

（正直、必要性を感じない...）

作るとすればクロスプラットフォームで。

今後の見通し

今後の見通し

ツールの実装

中間発表(4)ではお披露目 【必至】

実験とアンケート調査

これはツールが出来上がらないことには。。。

論文執筆

実験とアンケート調査が終わらないことには。。。

これぞ「クリティカルパス」orz

さいごに

実験・アンケート協力のお願い

歳末の多忙な時期にお願いすることになるやもしれません。

どうか温かい心で以ってご協力を賜りますよう、

宜しくお願い申し上げます。