Semat

～ Software Engineering Method and Theory ～

配布資料：Vision Statement 全文日本語
http://blogs.itmedia.co.jp/hiranabe/2010/03/semat-dd8b.html

SEA Forum on 29th, Nov. 2010

株式会社チェンジビジョン
平鍋健児

自己紹介

- ㈱永和システムマネジメント
  - 本社：福井県福井市、支社：東京(2002-)
  - Ruby と Agileを使ったシステム開発
- 株式会社チェンジビジョン
  - 本社: 東京
  - astah*(JUDE) で見える化
- 平鍋健児
  - リアルタイム，CAD, オブジェクト指向の実践
  - UMLエディタJUDE/astah*の開発
  - アジャイルプロセス協議会、副会長
  - 翻訳、XP関連書籍、『リーン開発の本質』等多数。
  - 2008 Gordon Pask Award Recipient for contributions to Agile practice

アジェンダ

- SEMAT とは何か?
- Call for Action
- Signatories
- The Vision Statement

- これまでの軌跡
- 他の意見

SEMAT

- Software Engineering Method and Theory
- *Semat seeks to develop a rigorous, theoretically sound basis for software engineering practice, and its wide adoption by industry and academia.*

Call for Action(1/2)

ソフトウェア工学は未成熟なプラクティス(immature practices)によって、重大な阻害(gravely hampered)を今日受けている。例えば、具体的には以下のように:

- 言葉の流行が、工学の一分野というよりファッション業界のようだ。
- しっかりした広く受け入れられた、理論的基礎の欠如。
- 非常に多くの方法論(methods)とその派生。またそれらの違いがほとんど理解されずに作為的に強調されている。
- 信頼できる実験的評価(experimental evaluation)と妥当性確認(validation)の欠如。
- 産業界の実践(industry practice)と学界の研究(academic research)の乖離。

Call for Action(2/2)

私たちは、ソフトウェア工学を堅固な理論および検証された原則とベストプラクティスを基礎として、再建するプロセスを支援する。そのプロセスは、以下の特徴を備えている。

- 広く合意された要素からなる、
  特定用途に拡張可能なカーネルを含み、
- 技術の問題と人の問題の両方を扱い、
- 産業界、学界、研究者そして、ユーザに支援され、
- 要求とテクノロジの変化に応じて追随できるような拡張性を備えている。

Vision Statement

- Purposes and scope(目的と範囲)
- The vision(ビジョン)
- The kernel(カーネル)
- The goals(ゴール)
- The principles(原則)
- One-year milestones(1年のマイルストン)
- Appendices
  - 「定義」、「理論」、「汎用要素」、「カーネル言語」、「アセスメント」

SEMAT

TWiki > Main Web > WebHome (2010-06-03, CarloAFuria)

Software Engineering Method and Theory

Signatories:

Pekka Abrahamsson	Scott Ambler	Victor Basili	Jean Bézivin
Dines Bjorner	Barry Boehm	Alan W. Brown	Larry Constantine
Steve Cook	Bill Curtis	Donald Firesmith	Erich Gamma
Carlo Ghezzi	Tom Gilb	Robert L. Glass	Ellen Gottesdiener
Martin Griss	Sam Guckenheimer	David Harel	Brian Henderson-Sellers
Watts Humphrey	Ivar Jacobson	Capers Jones	Philippe Kruchten
Robert Martin	Stephen Mellor	Bertrand Meyer	James Odell
Meilir Page-Jones	Dieter Rombach	Ken Schwaber	Alec Sharp
Richard Soley	Edward Yourdon		

SEMAT Signatories (1/4)

- Scott Ambler
  『アジャイルモデリング』著者。アジャイルデータ。現在IBM。
- Victor Basili
  GQM アプローチによるプロセス改善。現在はフラウンホーファー。
- Barry Boehm
  COCOMO 見積もりモデル、「変更コストは指数関数的に増加する」。『アジャイルと規律』にて、はじめて計画駆動とのバランスと「アジャイルのスイートスポット」を言った人。
- Larry Constantine
  ヨードンとともに、構造化設計から、コヒージョンとカプリング(凝集度と結合度)、という概念を導いた人。現在はユーザエクスペリエンス。

SEMAT Signatories (2/4)

- Erich Gamma
  Eclipse/IBM Jazz のリード。デザインパターンを書いたGoFの一人。Kent Beck とともに、JUnitを最初に開発。「テスト感染」という言葉。IBMスイス。
- Tom Gilb
  Evo という「世界初のアジャイル方法論者」。@imtomgilb。
- David Harel
  状態遷移図の開祖。状態遷移図のことを、「ハレル図」っていうことを知っているか？彼は、ユースケースのことを大粒度の状態、とも呼んでいる。
- Robert L. Glass
  『ソフトウェア・クリエイティビティ』、『ソフトウエア開発 55の真実と10のウソ』

SEMAT Signatories (3/4)

- Watts Humphrey(2010, 10月他界)
  カーネギメロン大学（SEI: Software Engineering Institute）。ソフトウェア品質の父、とも呼ばれる。成熟度モデルCMM, TSP, PSPの祖。
- Capers Jones
  見積もりといえばこの人。FP(ファンクションポイント)法の祖。
- Ivar Jacobson
  UMLを作った3アミーゴの１人。OOSE開発方法論。ユースケースの開祖。「ソフトウェアプロセスの話はもうたくさんだ！」と、RUPを離れて、Essential UP をプラクティスベースで提供。
- Philippe Kruchten
  クルーシュテン博士。RUPの祖。アジャイルを工学的にバランスよく捉えている人の一人。（アジャイルは単に廃れつつある流行語なのか）
- Robert Martin
  90年代 C++ Report 編集長。Fitnesse 開発者。ソフトウェア設計原則 SOLID。『オブジェクト指向開発の奥義』@unclebobmartin

SEMAT Signatories (4/4)

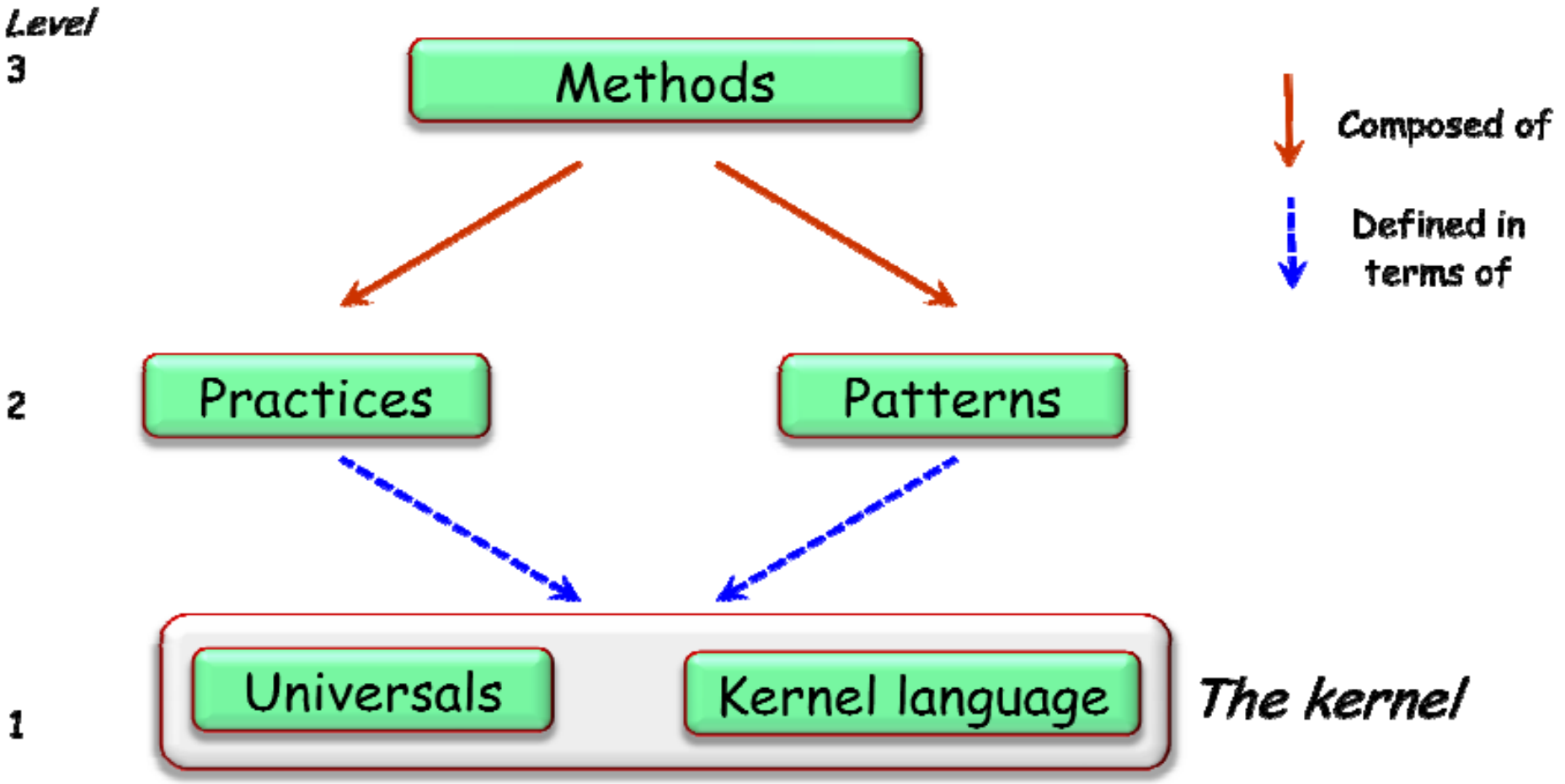
- Stephen Mellor
  シュレイヤ・メラー法。実行可能UMLによって、アジャイル宣言の一人。
- Bertrand Meyer
  大著『オブジェクト指向入門』。契約による設計(Design by Contract)。オブジェクト指向プログラミング言語、Eiffelを設計した。
- Dieter Rombach
  ロンバック博士。現在フラウンホーファーのエグゼキュティブディレクタ。Experimental Software Engineering
- Ken Schwaber
  アジャイル方法論Scrumの父。
- Richard Soley
  OMGの会長。

The Vision Statement

ビジョン(The vision)

- Semat のビジョンは、2段構成である。
- 付録も含めてこの文書の以下の章に書かれたすべてのゴールを達成すること。
- 人々が、自分たちの現在、未来のプラクティス(practices)、パターン(patterns)、メソッド(methods)を記述(describe)し、それらを組合せ(compose)、模倣(simulate)、応用(apply)、比較(compare)、評価(evaluate)、計測(measure)、教育(teach)、研究(research)できるようなプラットフォーム(カーネル)を作る。

カーネル(The kernel)



カーネル(The kernel)

- カーネル(kernel):ユニバーサル (汎用要素: universal)とカーネル言語(kernel language)(図1 のLevel 1)
- プラクティス(practices)とパターン(patterns):カーネル言語によって定義される(Level 2)
- メソッド(methods):プラクティスとパターンの組み合わせによって定義される(Level 3)

ゴール(The goals)

5つのトラックから構成される。

- 定義(Definitions)： ソフトウェア工学、およびその領域のその他本質的なコンセプトを定義する。
- 理論(Theory)： 本質的な助力を提供する理論(特に数学からの)。
- ユニバーサル(Universals)： Sematカーネルに組み込むべき、ソフトウェア工学の汎用要素を特定する。
- カーネル言語(Kernel language)： ユニバーサル、プラクティス、パターンを記述する言語を定義する。
- 評価(Assessment)： ソフトウェア工学のプラクティス、理論を評価する手法。(Semat自体の評価も含む)

原則(The principles) for kernel

- 品質(Quality)。主目標はソフトウェアプロダクトとプロセスの改善。
- シンプルさ(Simplicity)。カーネルには本質的なコンセプトのみを含む。
- 理論(Theory)。カーネルは堅固で厳密な理論的基礎に築かれる。
- 現実性とスケーラビリティ(Realism and scalability)。カーネルは実践的なプロジェクト(大規模プロジェクトを含む)に適用可能で、そこで検証可能な手法でなければならない。
- 正当性(Justification)。すべての提案は、明確な論拠によって正当化されなければならない。
- 反証可能性(Falsifiability)。すべての主張は、実験的な評価と反論を受けなければならない。
- 先見性(Forward-looking perspective)。前世代に起こった方法論の淘汰を考慮に入れつつも、完全な互換性には縛られない。
- モジュール性(Modularity)。プラクティスとパターンはカーネルを使って定義され、それぞれの組織のニーズに合うように組み合わせたり調整したりできる。
- 自己改善(Self-improvement)。カーネルは自身の進化を可能にする仕組みを搭載しなければならない。

原則(The principles) for process

- オープン性(Openness)。カーネルの開発においては、Semat活動のメンバーからの適切な形式の示唆は、すべて考慮対象とされなければならない。
- 公平性(Fairness)。貢献するすべてのアイディアは、功績として評価されなければならない。どんな側面も、特定のステークホルダやコミュニティの利益に偏って設計されてはならない。
- 目的性(Objectivity)。アイディアは、前もって明確に定義された目的性の判断基準によって評価されなければならない。
- タイムリー性(Timeliness)。進捗と結果をデリバリするために、締め切りを設けてそれを監視しなければならない。

付録

- A.1.1　ソフトウェア工学とは何か？（Definition of Software Engineering）
- 理論(Theory)
  - 「確率」、「統計」、「待ち行列理論」
  - カテゴリ理論（オブジェクト指向の認識論）
  - 論理学：オートマトン理論、モデルチェッキングが動機
  - 経済学、社会学、心理学：組織、マネジメント、コミュニケーション、協調、その他ソフトウェアの人間的な分野

学会と産業

- ソフトウェアにおいて、実践者と学界との溝は他の工学分野よりも深刻である。例えば電気エンジニアがマックスウェル方程式をとりあげて、実践と無関係な純粋に学問的な研究だ、などと言ったりすることは想像し難い。しかし、ソフトウェアのエンジニアとマネジャーが、現場にとっても重要な理論的貢献(例えば形式検証技術)に関して、そのようなコメントをすることはよくある。さらには、実践者がそういった手法のことをまったく聞いたこともない、というケースさえある。
- アカデミック側にも、まったく罪がないわけではない。研究のいくつかは、科学的に挑戦的な課題であっても産業界の関心と関連付けることが難しいものもある。
- 2つのキャンプの交流はここ数年で改善している。形式手法を例にとるとITの数多くの分野で利用例が増えてきている(一見形式手法に見えないやり方で使われている例もある)。
- ソフトウェア工学が他の工学分野のレベルにまで成長し、かつ、学界と産業界の溝を完全には取り除かないように(研究と応用は別の役割であるから望ましいことではない)しながら、理論と実践の健全な関係を構築することが、Sematのゴールの1つである。

その後の経緯

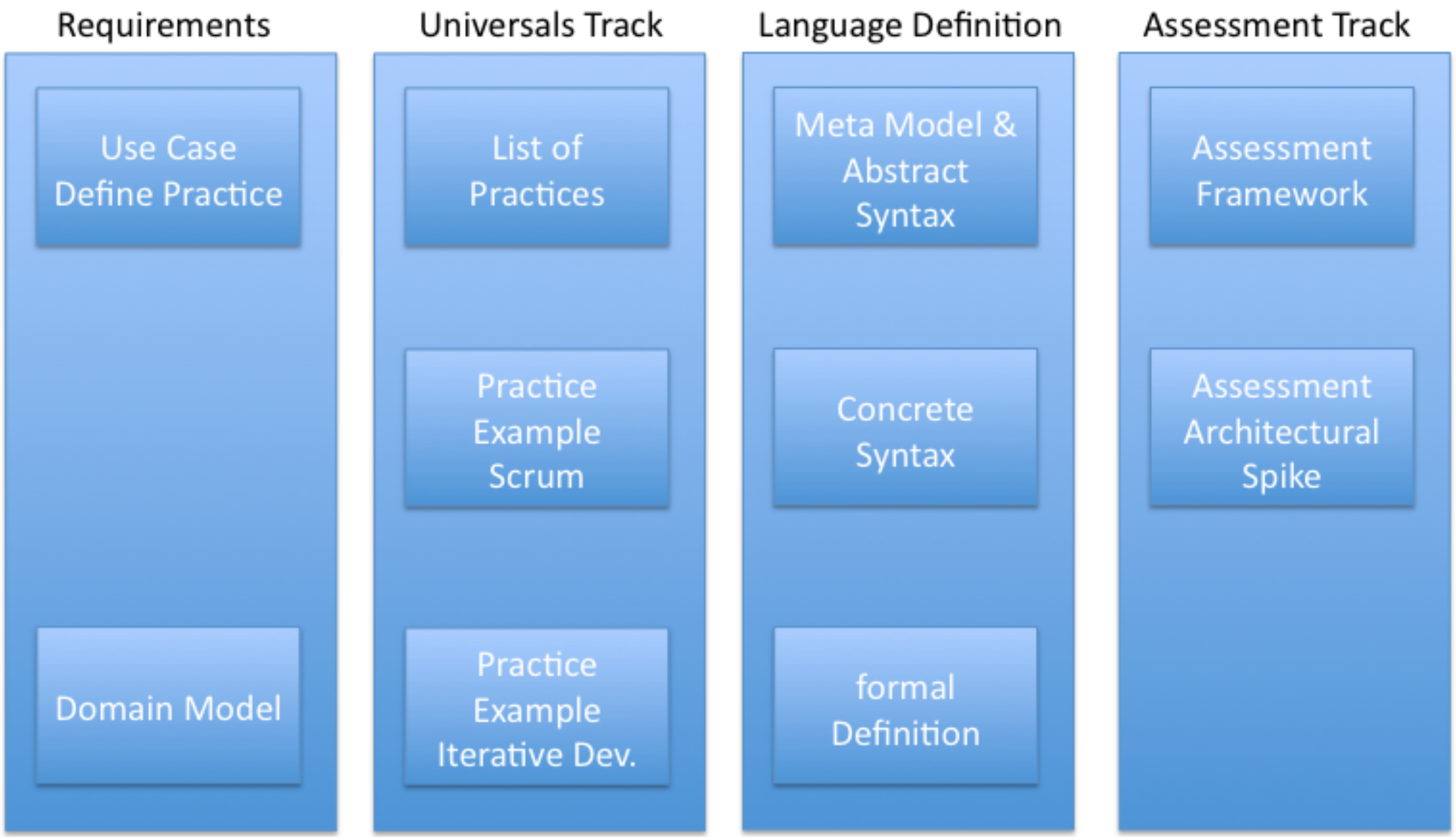
これまでの経緯

- [2010-10-20]
  Semat Moved under OMG standardization process
- [2010-9-29,10-1]
  The 3rd Semat workshop in Milan Italy
- [2010-06-13,14]
  The 2nd Semat workshop in Washington D.C.
- [2010-03-17,18]
  The 1st Semat workshop in Zurich
- [2010-02-16]
  The SEMAT Vision Statement by Ivar, Bertrand, and Richard.

各トラックの状況

- Requirement Track
  - Use Case to contextualize and provide scope of Semat.
  - 20 users(8 of which are Practitioner), 102 Things, Activities.
  - 27 scenarios elaborated
  - 5 use cases, and Domain Model, architecture spike for Domain Model to emerge
- Universals
  - 7 identified: Stakeholder Community, Software System,Team, Opportunity, Requirements, Work ,Practice, Method
  - 232 Practices, top 2 = “Scrum” and “Iterative Development”
- Kernel Language
  - 14 use cases, initial priority: “define practice”, “validate practice”, “compose practice”
  - “defining a software engineering practice“
- Theory
  - two kinds of theories:
  - mathematic/logics based that use formal languages, algebraic systems;
  - empirical theories that use statistics, psychology, sociology, etc.

Kernel の扱うテーマ



他の意見

Definition of Software Engineering from Wikipedia (= SWEBOK)

"Software engineering is the application of a **systematic, disciplined, quantifiable approach** to the development, operation, and maintenance of software, and the study of these approaches; that is, the application of engineering to software"

Definition of Software Engineering
Tom Gilb

"Software engineering is the engineering discipline of enabling and motivating software systems to deliver **a balanced set of values**, directly or indirectly, **to a balanced set of stakeholders**, throughout their lifecycle…

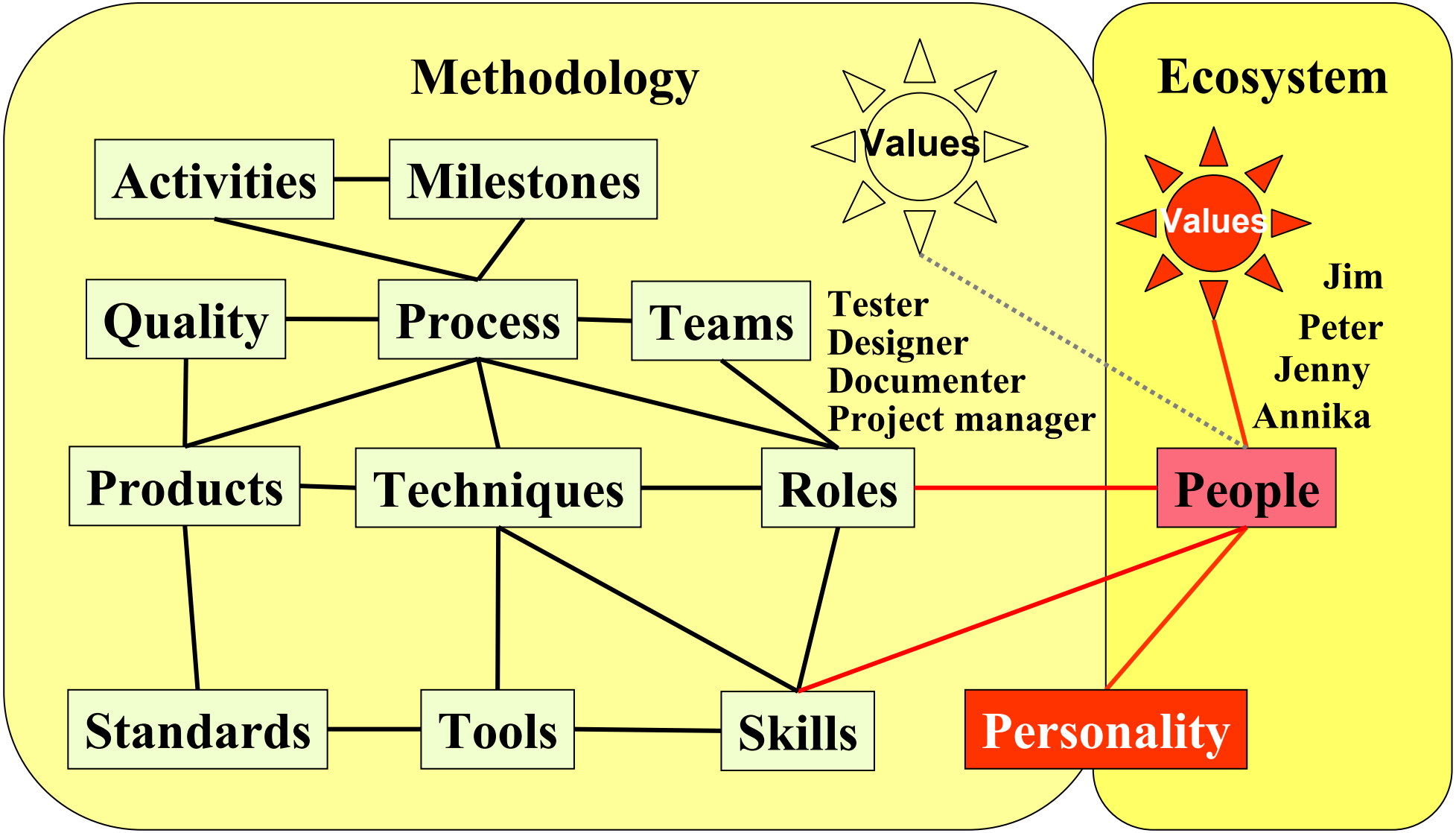
The concept 'balanced set of value' (above) is …The concept 'a balanced set of stakeholders' (above) is …"

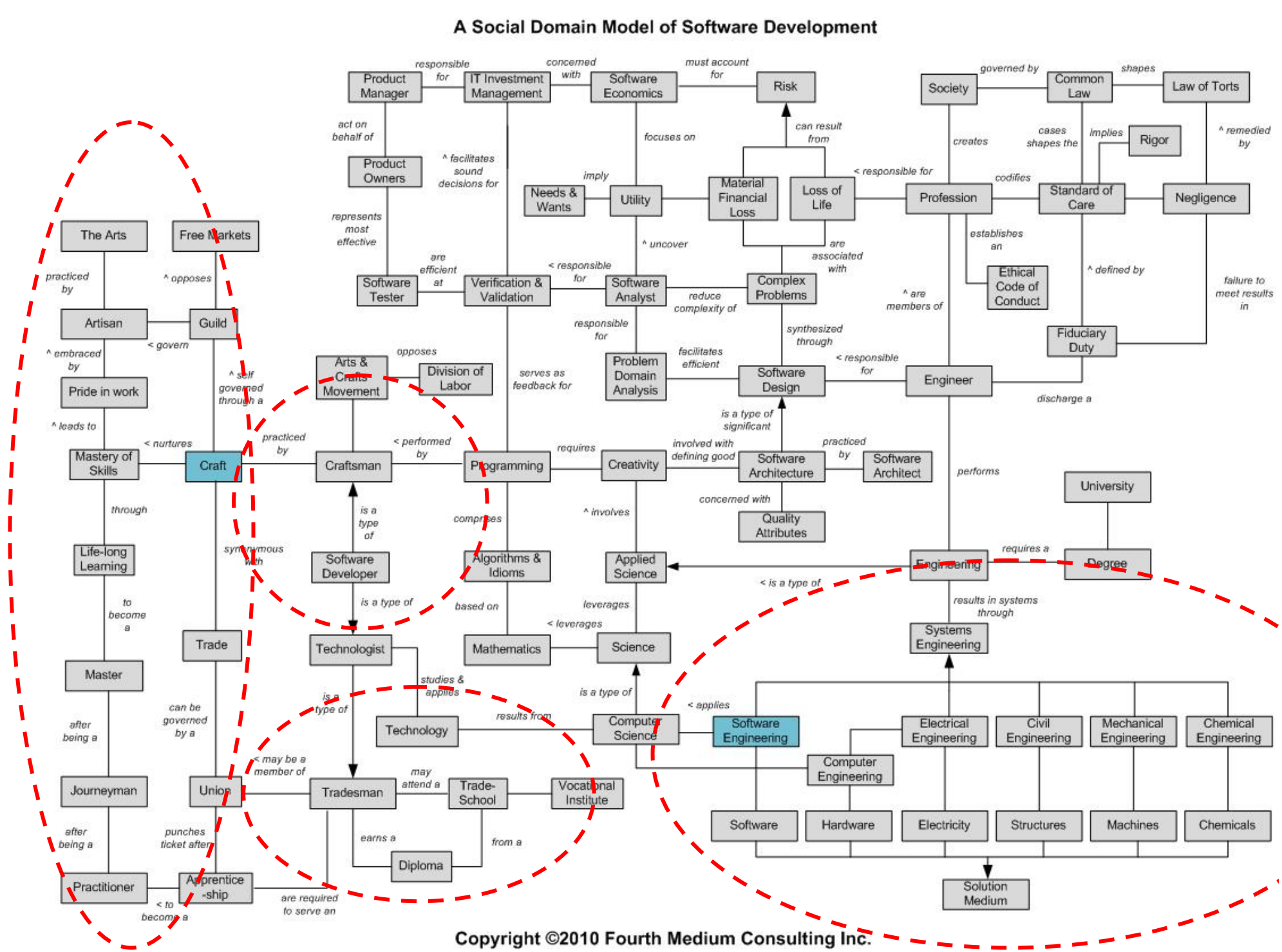
The end of software engineering and the start of economic-cooperative gaming - Alistair Cockburn

"Software development is not "naturally" a branch of engineering. It was proposed in 1968…The term "software engineering" fails a crucial test, that of suggesting good actions to the busy practitioner. …"

"Viewing software development as a **"series of resource-limited, cooperative games of invention and communication"** meets the objectives …"

But people are stuffed full of personality





Mark Kennaley http://www.fourth-medium.com/wordpress/?p=150