エンタープライズ統一プロセス (EUP)入門

A Ronin International, Inc. White Paper

Scott W. Ambler
Senior Consultant
Ronin International, Inc.
(株)オージス総研　訳

This Version: February 8, 2005

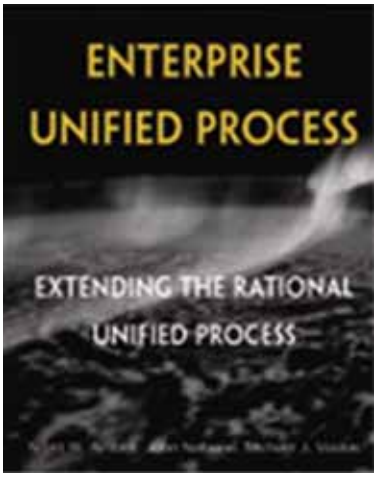
Copyright 1999-2005 Scott W. Ambler, Ronin International, Inc.
Visit www.enterpriseunifiedprocess.com

謝辞

Ronin International 社の John Nalbone と Mike Vizdos のソフトウェアに関する洞察に対して感謝します。両者は、複数の組織に EUP を導入するための手助けを行い、EUP の発展と開発に積極的に関与してきました。

本ホワイトペーパの元本

本ホワイトペーパは、Scott W. Ambler, John Nalbone, Michael Vizdos の共著で 2005 年 2 月に Prentice Hall PTR から出版された”*The Enterprise Unified Process: Extending the Rational Unified Process* ”を要約したものです。.



日本語訳に関する注意点

暫定的な訳ですので、今後訳語の変更などを行う可能性があります。その点につき、ご了承をお願いいたします。

Copyright 1999-2005 Scott W. Ambler, Ronin International, Inc.
Visit www.enterpriseunifiedprocess.com

目次

1. ラショナル統一プロセス (RUP) 1

2. エンタープライズ統一プロセス(EUP) 3

2.1 運用及びサポート作業分野 5
2.2 エンタープライズ作業分野 6
2.3 稼動フェーズ 7
2.4 引退フェーズ 7
2.5 既存の作業分野の拡張 7

3. EUPメーリングリスト 8

4. 終わりに 9

5. 参照文献/推奨文献 10

5.1 WEB上の情報源 10
5.2 刊行された情報源 10

6. 著者について 13

Copyright 1999-2005 Scott W. Ambler, Ronin International, Inc.
Visit www.enterpriseunifiedprocess.com

ソフトウェア開発プロセスは、人々がソフトウェアや関係する成果物(計画、ドキュメント、モデル、コード、テストケース、マニュアル等)を開発し、保守するために用いる複数のプロジェクトフェーズ、ステージ、手法、テクニック、プラクティスから成ります。本ホワイトペーパでは、ラショナル統一プロセス(RUP) (Kruchten 2004)を概観し、RUP をどのように拡張することによりエンタープライズ統一プロセス(EUP)が形成されたかについて説明します。

1. ラショナル統一プロセス (RUP)

ラショナル統一プロセス(RUP) (Kruchten 2004)は、Rational 社（現在は IBM 社 Rational 事業部）、つまり、いまや業界標準のモデリング表記法となった統一モデリング言語(UML)を作り上げたのと同じ人たちの努力の成果として作成されました。RUP の中核部分には Objectory Process があります。これは何年も前に Ivar Jacobson の Objectory 社の組織を併合したときに Rational 社が獲得した製品やサービスの 1 つです。Rational 社は、自社や、買収した、あるいはパートナー関係を結んだツール会社のプロセスによって Objectory を拡張し、RUP の初期バージョン(5.0)を作成しました。これは 1998 年 12 月に公式にリリースされています。

図 1は現在のRUPのライフサイクルを表したものです。4 つの逐次的なフェーズと、9 つの基本となる作業分野(以前はワークフローと呼ばれていたもの)から構成されています。図の下の部分を見ると、RUPを通じて実施されるどの開発サイクルも、Rational社のいう反復というものに整理しなければならないことが分かります。私なら反復というよりインクリメントという言葉の方が適していると思いますが、基本的な考え方は、それぞれの反復の終わりに、ユーザの人々に使ってもらうことが可能な実行可能ファイルを内部的に作成するということです。これによって、開発者と顧客とのコミュニケーションが向上し、プロジェクトのリスクを減らすことができます。もう 1 つ、リスク削減手法としてRUPに組み込まれているのは、各フェーズの終わりに先に進むか取りやめるかの決断を下すべきだという考え方です。もしプロジェクトが失敗するのであれば、ライフサイクルのできるだけ早い段階でやめるべきでしょう。これは、失敗率が 65%から 85%にもなる業界では重要な考え方です(www.standishgroup.com)。.

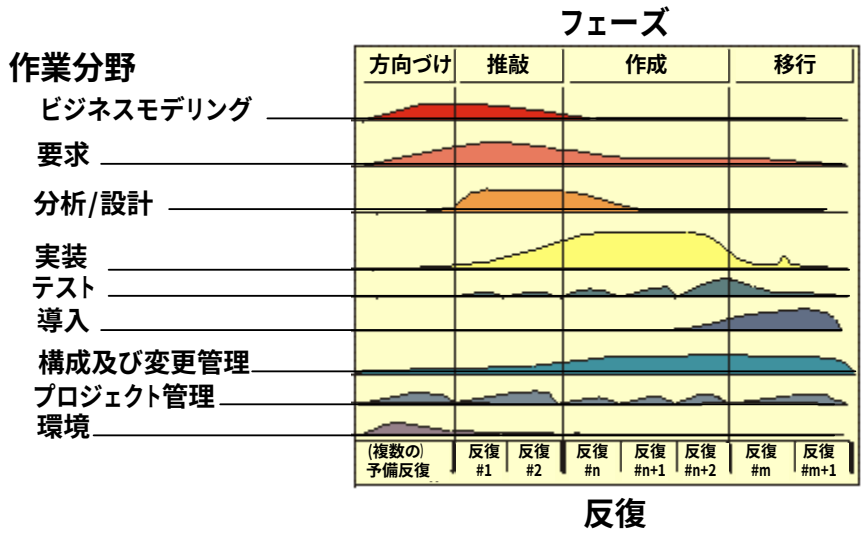


図 1 統一プロセスの初期ライフサイクル

方向づけフェーズ(www.ambysoft.com/inceptionPhase.html)では、プロジェクトの開発範囲を定義し、システムの開発企画書を定めます。ソフトウェアの初期のユースケースを洗い出し、主

Copyright 1999-2005 Scott W. Ambler, Ronin International, Inc.
Visit www.enterpriseunifiedprocess.com

要なものについては概略を記述します。ユースケースとは、システムの機能要求を定義するための業界標準の手法で、従来使われてきた要求文書に比べると非常に生産性が高くなっています。製品の特性ではなく、ユーザにとって何が価値を持つかに着目しているからです。初期段階のリスク評価、見積もり、プロジェクトのスケジュールなど、基本的なプロジェクト管理のための文書は、この方向づけフェーズで作成が始まります。もうお分かりだと思いますが、このフェーズの主要なタスクには、ビジネスモデリングや要求関連の作業の他、ツールの選択やプロセスのカスタマイズといった環境の初期定義も含まれます。

推敲フェーズ(www.ambysoft.com/elaborationPhase.html)では、問題領域の詳細な分析と、プロジェクトの基礎となるアーキテクチャの定義を主に行います。ユースケースだけであらゆる要求を定義することはできないので(Ambler 2001)、補足仕様書という納品物を定義してシステムの機能外要求をすべて記述します。このフェーズでは、方向づけフェーズで作り始めた初期の管理文書をもとに、作成フェーズ用の詳細なプロジェクト計画も作成します。

作成フェーズ(www.ambysoft.com/constructionPhase.html)は、通常は(残念ながら)ソフトウェアプロジェクトの中心となるフェーズであり、アプリケーションの詳細設計を行い、それに対応するソースコードを開発します。プロジェクトがこのフェーズに主に注力することを「残念ながら」と言ったのは、多くの場合、組織がその前の 2 つのフェーズに十分なリソースをつぎ込まないため、ユーザのニーズを満たすソフトウェアの開発を成功させるための基礎ができあがっていないからです。このフェーズの目標は、ユーザの人々に引き渡すソフトウェアおよびサポート文書を作成することです。

移行フェーズ(www.ambysoft.com/transitionProductionPhase.html)の目的は、システムをユーザの人々に納品することです。ソフトウェアのベータ版をユーザにリリースすることもよくあります。これは一般にパイロットリリースと呼ばれるもので、利用者全体にリリースする前に、システムを少数のユーザに使ってもらうというものです。大きな欠陥は、このフェーズで見つけて、可能な限り対応します。最後に、これまでの作業が成功であったかを評価し、さらにシステムを拡張するための開発サイクルやインクリメントが必要かどうかを判断します。

RUP には、以下のような長所があります。

1. 反復型、要求駆動、かつアーキテクチャベースの開発アプローチを取るなど、しっかりとしたソフトウェア工学の原則にもとづいている。
2. 各反復の最後に動くプロトタイプを作成する、各フェーズの最後に進むかやめるかの判断を下すなどのメカニズムがあり、管理者から開発プロセスが見えやすくなっている。
3. Rational 社が、これまでも現在もラショナル統一プロセス製品に十分な投資をし続けている。これは RUP を HTML ベースで記述したもので、各組織のニーズを完全に満たすように仕立てて使用できる。

同時に RUP には以下のような短所もあります。:

1. 開発プロセスでしかない。現在の RUP のバージョンでは、ソフトウェアプロセス全体をカバーしていない。図 1を見ると、システムが稼動段階に入ってからの運用やサポートといった概念が明らかに欠けている。また、いつかシステムを引退させるという概念も含まれていない。
2. RUP では、組織全体のアーキテクチャのモデリングなど、複数システムのインフラストラクチャ開発作業を明示的にサポートしていない。そのため、組織内で大規模な再利用をすることができない。
3. 多くのベテラン開発者にとって、反復的なライフサイクルは耳慣れないものであり、受け入れるのが困難である。図 1のようにライフサイクルを表現しても、まったくこの問題の解決になっていない。統一プロセスの反復的な性質は、長所でもあり短所でもある。

Copyright 1999-2005 Scott W. Ambler, Ronin International, Inc.
Visit www.enterpriseunifiedprocess.com

4. もともとラショナル社は、ツールを中心にこのプロセスを開発したため、できあがったプロセスは、今日の開発者が抱える複雑なニーズをおそらくまだ十分に満たすことができていない。すべてを自動化することはできないし、できるとしても、ラショナルは完全なツール群を持っていない。たとえば、ラショナル社はヘルプデスク製品を販売していない。ラショナル社はマーケティングを中心に進んでいる会社なので、ソフトウェアプロセスの中でもツールを販売していない部分については動機づけがほとんどない。システムのライフサイクルのうち、きわめて明らかな部分が RUP に欠けている理由は明白である。

2. エンタープライズ統一プロセス(EUP)

それでは、典型的な組織の実際のニーズに合うようRUPを拡張するにはどうすればよいのでしょうか。まず最初に、RUPの対象範囲を広げて、開発プロセスだけでなくソフトウェアプロセス全体を含める必要があります。組織には、現在管理しているソフトウェアプロジェクトが、おそらくいくつもあるでしょう。現在稼動(production)段階として運用・サポートしているシステムもいくつかあるはずです。ほとんどの組織で実際に重視しているのは、1つのプロジェクトの開発ではなく、複数のシステムの開発や運用やサポートや保守です。つまり、運用やサポートや保守の作業のためのプロセスをRUPに追加する必要があるということになります。次に、今日の組織のニーズを十分に満たすには、RUPでプロジェクトのポートフォリオ管理をサポートする必要もあります。これは他のプロセスで、プログラム(programme)、マルチプロジェクト、インフラストラクチャ管理、エンタープライズ管理などと呼ばれているものです。この2つのステップを加えることで、Larry Constantineと私がCMP Booksの統一プロセスシリーズ[1](Ambler & Constantine 2000a; Ambler & Constantine 2000b; Ambler & Constantine 2000c; Ambler & Constantine, 2002)で紹介した、拡張版の統一プロセスライフサイクルができあがります。このライフサイクルが、統一プロセスを発展させたインスタンスであるエンタープライズ統一プロセス(Enterprise Unified Process: EUP)です。

[1] この書籍シリーズは、Software Development (www.sdmagazine.com)誌に掲載されたRUPを強化するために利用できるテクニックに関する先端的な記事を集めたものに新たな内容を追加したものです。これらの記事で説明されたテクニックのいくつかはRUPの新バージョンに盛り込まれた点を私はうれしく思っていますが、RUPの今後のバージョンに私達がこの書籍を執筆する時点で予想したことがどの程度盛り込まれていくか見ていくのは非常に興味深いことでしょう。但し、この書籍に執筆されたことの多く、特にエンタープライズ管理作業分野と運用及びサポート作業分野はRUPに含まれていません。

Copyright 1999-2005 Scott W. Ambler, Ronin International, Inc.
Visit www.enterpriseunifiedprocess.com

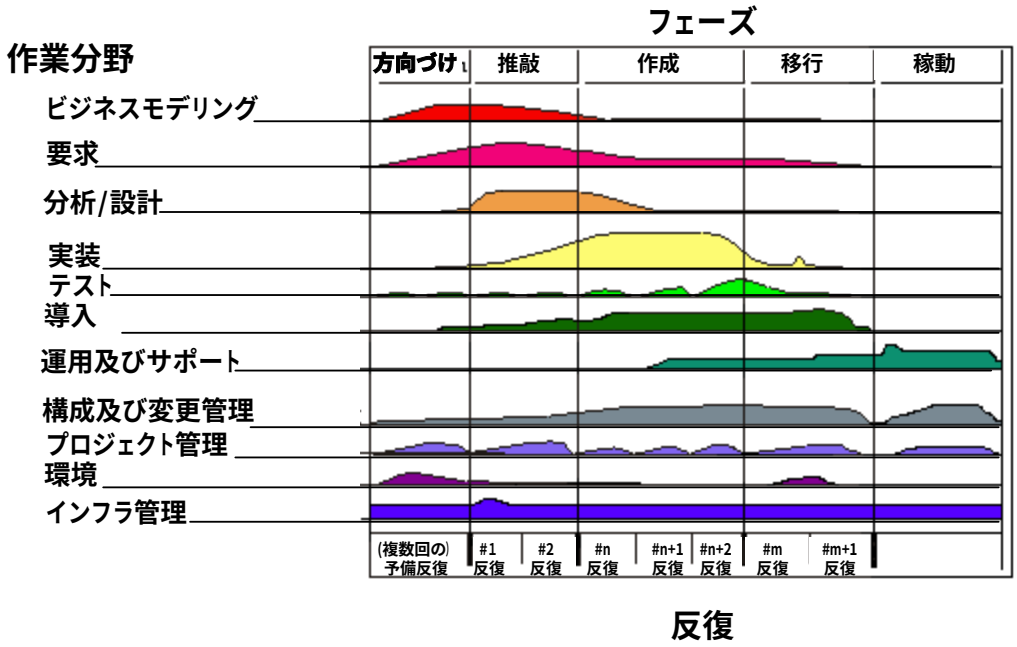


図 2　エンタープライズ統一プロセス (EUP)で拡張されたライフサイクル (v1.0).

2002 年 11 月には、システムが最終的には稼動状態から取り除かれることを反映するために、図 3に示されるように引退フェーズ(retirement phase)を含むようにライフサイクルを拡張しました。さらに、インフラ管理作業分野をエンタープライズ管理作業分野に名称を変えました。それは、インフラ管理作業分野だと多くの人が技術的なことだけを取り扱うと考えがちだったからです。

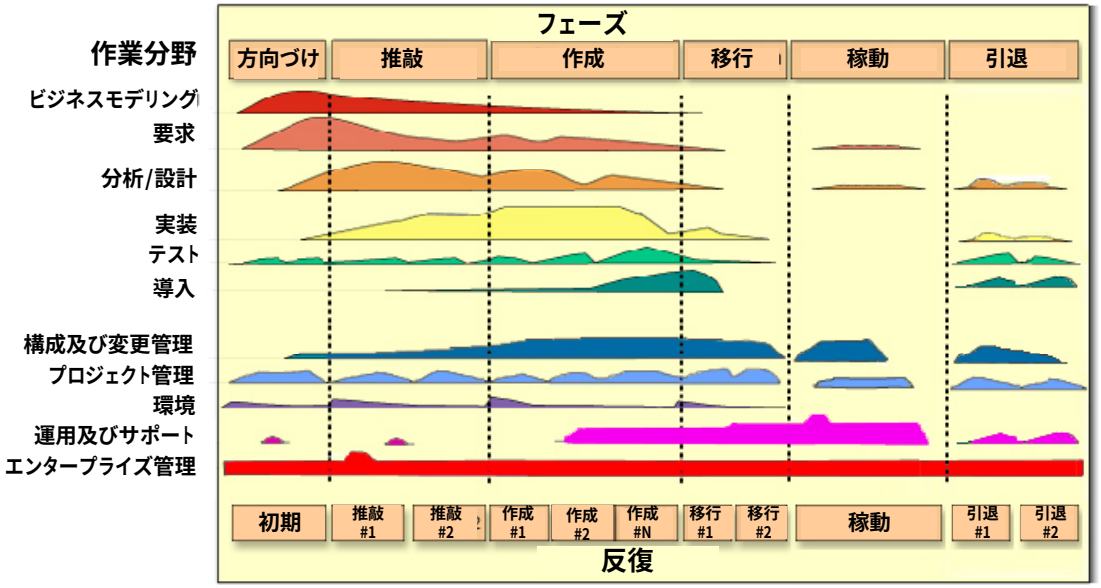


図 3　EUP の改訂されたライフサイクル (v2002).

Copyright 1999-2005 Scott W. Ambler, Ronin International, Inc.
Visit www.enterpriseunifiedprocess.com

2004 年の春、Mike Vizdos, John Nalbone, 私の 3 名は、図 4に示されるようにエンタープライズ管理作業分野を 7 つの作業分野に分割しました。また、作業分野全体を開発、サポート、エンタープライズの 3 作業分野群に整理しました。

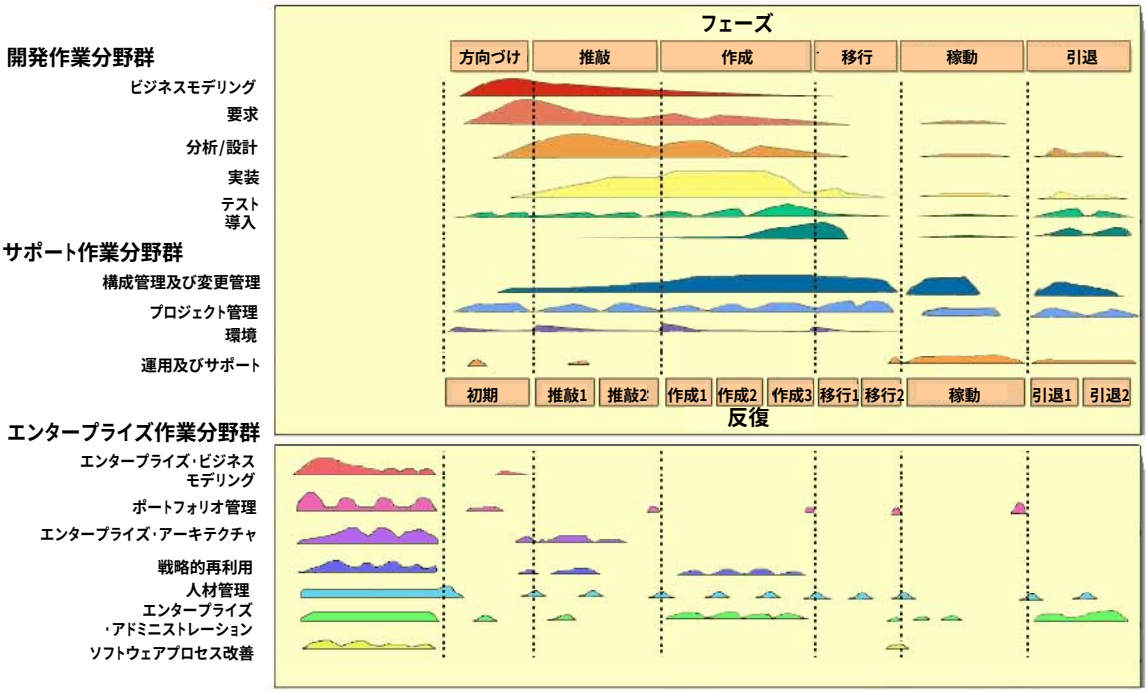


図 4　EUP のライフサイクル(v2004).

2.1　運用及びサポート作業分野

運用及びサポート作業分野 (Operation & Support discipline)の目的は、まさに名前が示す通り、ソフトウェアの運用とサポートです。運用もサポートも複雑な作業であり、そのプロセスを定義する必要があります。他の作業分野と同様に、この作業分野も複数のフェーズにまたがって実施されます。作成フェーズや、早ければ推敲フェーズの段階で、運用とサポートのための計画や文書やトレーニングマニュアルを作成する必要があるでしょう。移行フェーズでは、引き続きこれらの成果物を開発し、テスト結果にもとづいて作り直し、運用やサポートの担当者がソフトウェアを効果的に使えるようトレーニングを行います。稼動フェーズでは、運用担当者がソフトウェアを動作させ続け、必要なバックアップやバッチジョブを実行します。また、サポート担当者はソフトウェアを使うユーザの人々に対応します。さらに、引退フェーズでレガシーシステムの稼動を止めるときにも、この作業分野が関係します。

この作業分野は、基本的にはOOSPのリリース・ステージとサポート・ステージ(Ambler 1998b)、OPEN プロセスの実装計画作業とシステムの利用作業(Graham, Henderson-Sellers, and Younessi 1997)、ISO 12207 の*Operational Support* and *Dispensation* phases (www.iso.org)の一部を包含するものです。

Copyright 1999-2005 Scott W. Ambler, Ronin International, Inc.
Visit www.enterpriseunifiedprocess.com

2.2 エンタープライズ作業分野群

EUP には、組織が IT において成功を収めるために解決しなくてはならないシステム横断的な問題に対処するための７つのエンタープライズ管理作業分野が含まれています。これらの作業分野は、以下の通りです。:

1. **エンタープライズ・ビジネスモデリング (Enterprise Business Modeling)**. この作業分野は、企業のビジネス構造とプロセスを検討することを目的とする。エンタープライズ・ビジネスモデリングは、問題及び自動化すべき候補領域を特定するのに役立つ。
2. **ポートフォリオ管理 (Portfolio Management)**. 組織は、個別のアプリケーションではなく、しばしばプログラム群と呼ばれる全体としてより良く管理されるようなアプリケーション群を有している。この作業分野により、自組織のソフトウェア・ポートフォリオ全体もポートフォリオ内の個別のプログラムも追跡したり、計画することが可能になる。そのようにすることで、新たな要求をもっと戦略的に実装したり、計画したりすることが可能になる。さらに、この作業分野により、異なるアプリケーションで同じ機能が実装されることを避けることができる。
3. **エンタープライズ・アーキテクチャ (Enterprise Architecture)**. この作業分野は、組織におけるアーキテクチャに関する問題全体に対処するためのものである。この作業分野には、アーキテクチャを定義するためのモデル、アーキテクチャの動作を示すためのプロトタイプと仕掛かったモデル、アーキテクチャを使いやすくするためのフレームワークが含まれる。
4. **戦略的再利用 (Strategic Reuse)**. この作業分野は、プロジェクト間で資産の開発や再利用を促進し、毎回新たに開発するのではなく、資産を再利用しながら高品質のアプリケーションをより迅速に開発することを目指すものである。さらに、既にテストされ、動くことが実証されている成果物を利用することにより、品質の向上も支援する。
5. **人材管理 (People Management)**. ソフトウェア開発の作業を管理するということには、プロジェクトプランやスケジュールを作成したり、発展させたりするよりもはるかに多くのことが含まれる。組織には人材が存在し、スタッフを管理したり、スタッフの間や他の人たちとの相互作用を管理する必要が求められる。この作業分野では、人材が良く作業をし、組織内のプロジェクトに成功理に貢献するように、人材を組織化したり、モニターしたり、コーチしたり、動機づけするためのプロセスを記述しています。
6. **エンタープライズ・アドミニストレーション (Enterprise Administration)**. この作業分野には、IT 組織の主要なインフラコンポーネントであるツール、プロセス、設備をセットアップしたり、管理する作業が含まれる。
7. **ソフトウェアプロセス改善 (Software Process Improvement)**. この作業分野では、組織内の複数のプロセスを管理したり、改善したり、支援する必要性に対処するものである。

これらの新しいエンタープライズ作業分野の実践は簡単ではありません。これらの作業分野は慎重に実装されるべきであり、現在のビジネス機能の阻害を最小限に留めなければなりません。これらの作業分野を官僚主義に陥らず実装することにより、これらの作業分野は IT 部門のオペレーションを効率化することにより、長期的には莫大な費用の削減を成し遂げることができるでしょう。

エンタープライズ管理作業分野と既存の RUP の作業との間にはある程度の重なりがあるものの、スコープが異なっているということを理解することは大事です。例えば、エンタープライズ・アーキテクチャ作業分野には、RUP の分析/設計作業分野中の作業が多数含まれています。エンタープライズ・アーキテクチャでは複数のシステムを横断するアーキテクチャを取り扱われるのに対して、分析/設計作業分野では単一のシステムのアーキテクチャが取り扱われます。同様に、エンタープライズ・ビジネスモデリング(EBM)とビジネスモデリング(BM)の作業につ

Copyright 1999-2005 Scott W. Ambler, Ronin International, Inc.

Visit www.enterpriseunifiedprocess.com

いて同じことがいえます。EBM はビジネス全体を広く、浅く見たものであるのに対して、BM は単一のシステムに関係するビジネスだけを見たものです。BM は、特定のシステムを開発するために必要になるように、特定の分野により深く関わるものになります。

2.3 稼動フェーズ

図 4に示されるように、EUP にはシステムが導入された以降のソフトウェアライフサイクルの一部を表す５番目のフェーズである稼動(production)が含まれます。フェーズ名から分かるように、その目的は、新しいバージョンで置き換えるか(バグフィックスなどのマイナーリリースから新しいメジャーリリースまで)、引退して稼動状態から外されるまで、ソフトウェアを稼動させ続けることです。このフェーズでは反復を行わない(あるいは見方によっては反復を 1 回だけ行う)ことに注意してください。このフェーズはソフトウェアの 1 つのリリースに対して適用されるものだからです。ソフトウェアの新しいリリースを開発して導入するには、もう一度 4 つの開発フェーズすべてを実施する必要があります。

2.4 引退フェーズ

引退フェーズ (retirement phase)の中心は、稼動状態からシステムを首尾よく取り除くことです。システムの稼動を止める理由はいくつかあります。

1. 必要がなくなったから。たとえば、政府の法律上の要求を満たすために稼動させたシステムなど。法律が廃止されれば、システムの必要はなくなる。
2. システムがリプレースされたから。たとえば、自社開発の人事システムを市販パッケージ(COTS)システムでリプレースするのはよくあることである。

引退フェーズの作業には次のようなものがあります。

1. 他のシステムとの結びつきを明らかにするための、既存システムの広範囲にわたる分析。
2. 他の既存システムが引退するシステムに依存しないようにするための、設計や開発のやり直し。この作業は一般に、独立したプロジェクトとして扱われる。
3. システムが引退して必要でなくなったり、あるいは操作されなくなった既存のレガシーデータの変換。通常は、データベースリファクタリング (www.agiledata.org/essays/databaseRefactoring.html) によって行う。
4. 引退するシステムで保守していたデータのうち、他のシステムで必要でないデータのアーカイブ作成。
5. 今後いつか必要になった場合に再インストールできるようにするための、削除対象ソフトウェアの構成管理(口で言うほどに簡単ではない)。
6. 対象システムを引退させたことにより、残ったシステムに問題が生じていないことを確認するためのシステム統合テスト。

2.5 既存の作業分野の拡張

図 2のEUPの拡張されたライフサイクルを統一プロセスの初期ライフサイクルを対比すると、いくつかの既存の作業分野が改訂されていることに気づくでしょう。まず、テスト作業分野が拡張され、方向づけフェーズにおける作業が追加されています。このフェーズでは、初期段階の高いレベルの要求を作成しますが、この要求は、ウォークスルーやインスペクションやシナリオテストといった手法によって検証することができます。OOSPの根底にある原理の中には、早い段階から頻繁にテストをするべきである、開発する価値のあるものはテストする価値がある、という 2 つがあります。エクストリーム・プログラミング(XP) (Beck 2000)は、これを一歩進めて、テストファーストのアプローチを取るように主張さえしているのです。私は自分の経

Copyright 1999-2005 Scott W. Ambler, Ronin International, Inc.
Visit www.enterpriseunifiedprocess.com

験から、テストはライフサイクルの早い段階に移動するべきだと考えています。また、テスト作業分野は、「フルライフサイクルオブジェクト指向テスト(Full Lifecycle Object Oriented Testing: FLOOT)」(Ambler 2004)[2]で強化することもできます。

2つめの変更は導入作業分野に対するものです。この作業分野を方向づけフェーズや推敲フェーズにまで伸ばしました。これは、(少なくともビジネスアプリケーションの)導入とは恐ろしい仕事であるという事実を反映するための変更です。レガシーデータソースのデータ変換作業は独立したプロジェクトとして行われることも多く、かなりの計画と分析、そして相当な工数が必要です。さらに、配置モデリングは、現在含められている分析と設計の作業分野ではなく、導入作業分野に含めるべきだと私は考えています。配置モデリングと導入計画は関連して行われるからです。導入計画は、方向づけフェーズという早い段階から開始して推敲フェーズおよび作成フェーズへと続けることができますし、そうするべきです。そして、それと並行して配置モデリングを行います。

環境作業分野は、稼動フェーズの環境を定義するのに必要な作業を含めるように変更されています。通常なら移行フェーズで行われる作業です。もともとの環境作業分野のプロセスはほとんど同じで、唯一変わっているのは、その対象とする範囲を開発環境だけでなく運用やサポートの環境まで広げなければならない点です。開発者と同じく、運用やサポートの担当者も、それぞれのプロセスや標準、ガイドラインやツールが必要です。そのため、このニーズを反映するため、カスタマイズしたり、開発したり、購入したりといったことが必要になる可能性があります。

構成及び変更管理の作業分野は、新しく追加された稼動フェーズまで範囲を広げて、導入したソフトウェアに対して提案された変更の影響を評価し、その変更をシステムの将来的なリリースに割り当てるのに必要な、変更管理のプロセスを含むようになっています。この変更管理のプロセスは、開発時よりもこのフェーズの方がたいていは公式に行われますが、それは既存のソフトウェアを更新したりリリースし直したりする作業が増えるためです。また、この作業分野は引退フェーズまで範囲が広がっていて、システムを稼動状態から取り除くという作業に関する構成管理の問題をサポートできるようになっています。

同様に、プロジェクト管理作業分野も新しい稼動フェーズおよび引退フェーズまで広げられ、その作業を管理するのに必要なプロセスを含むようになっています。プロジェクト管理作業分野には、新しい機能を追加すべきです。先に述べたように測定管理の作業は不十分なのですが、この分野について Rational 社は最近進歩を見せています。また、開発委託先の管理のためのプロセスも必要です。これは、開発作業の一部をアウトソーシングしたり、コンサルタントや受託業者を雇っている組織にとって、大きな要求です。トレーニングや教育の他にキャリア管理まで含めた人材管理の問題は、RUP ではほとんど触れられていません。プロジェクト管理は、プロジェクト計画を作成して発展させるという技術的な課題だけではありません。人員を管理し、その人員間、あるいは他の人たちとの相互作用を取り持つ必要もあります。

3. EUP メーリングリスト

エンタープライズ統一プロセス(EUP)について質問があれば、気軽にEUPメーリングリストに投稿してください。これは無料のサービスで、参加方法の詳細は

[2] Software Development誌(www.sdmagazine.com)に統一プロセスの記事(Ambler 1999b)を書き始めた時以来、IBM/Rational社はテスト作業分野を大幅に改善しました。私は、この改善に感銘を受けており、今後のさらなる改善に期待しています。

Copyright 1999-2005 Scott W. Ambler, Ronin International, Inc.
Visit www.enterpriseunifiedprocess.com

www.enterpriseunifiedprocess.com/feedback.html に書いてあります。EUPに興味があれば、あるいは現実に即したRUPの仕立て方(tailoring)に関して広く議論をしたければ、是非このリストに参加してください。

4. 終わりに

本ホワイトペーパでは、現実世界の典型的な組織ニーズに対応できるような統一プロセスのインスタンスとしてエンタープライズ統一プロセス(EUP)を紹介しました。EUP は、大規模開発の厳しいニーズに対応できるように仕立てることができます。ソフトウェア開発、保守、サポートは複雑な努力であり、その努力を成功させるためには優れた人材、優れたツール、優れたアーキテクチャ、優れたプロセスが必要になります。EUP は、無視すれば危険に陥るようなソフトウェアの危機を解決する大きな部分となります。

Copyright 1999-2005 Scott W. Ambler, Ronin International, Inc.
Visit www.enterpriseunifiedprocess.com

5. 参照文献/推奨文献

5.1 Web 上の情報源

Agile Data Home Page, www.agiledata.org

Agile Modeling Home Page, www.agilemodeling.com
邦訳（一部）：アジャイルモデリング(AM)日本語サイト, http://www.ogis-swe.jp/process/am-res/am/index.html

Enterprise Unified Process, www.enterpriseunifiedprocess.com

The OPEN Website. www.open.org.au

The Process Patterns Resource Page. www.ambysoft.com/processPatternsPage.html

Rational Unified Process. www.rational.com/products/rup

Software Engineering Institute Home Page. www.sei.cmu.edu

5.2 刊行された情報源

Ambler, S.W. 1998a. *Building Object Applications That Work*. Cambridge University Press. www.ambysoft.com/buildingObjectApplications.html.

Ambler, S.W. 1998b. *Process Patterns: Building Large-Scale Systems Using Object Technology*. Cambridge: Cambridge University Press. www.ambysoft.com/processPatterns.html.

Ambler, S.W. 1999a. *More Process Patterns: Delivering Large-Scale Systems Using Object Technology*. Cambridge: Cambridge University Press. www.ambysoft.com/moreProcessPatterns.html.

Ambler, S.W. 1999b. *Enhancing the Unified Process*. Software Development Magazine, October 1999. www.sdmagazine.com/documents/s=753/sdm9910b/9910b.htm

Ambler, S.W. (2001). Agile Modeling and the Unified Process. www.agilemodeling.com/essays/agileModelingRUP.htm

Ambler, S.W. (2002). *Agile Modeling: Effective Practices for Extreme Programming and the Unified Process*. New York: John Wiley & Sons. www.ambysoft.com/agileModeling.html
邦訳: スコット W. アンブラー, アジャイルモデリング : XP と統一プロセスを補完するプラクティス, オージス総研監訳, 翔泳社, 2003

Ambler, S.W. (2004). *The Object Primer 3rd Edition: Agile Model Driven Development with UML 2*. New York: Cambridge University Press. www.ambysoft.com/theObjectPrimer.html.

Ambler, S.W. (2005). *The Elements of UML 2.0 Style*. New York: Cambridge University Press. www.ambysoft.com/elementsUMLStyle.html

Copyright 1999-2005 Scott W. Ambler, Ronin International, Inc.
Visit www.enterpriseunifiedprocess.com

Ambler, S.W. & Constantine, L.L. (2000a). *The Unified Process Inception Phase*. Gilroy, CA: CMP Books. www.ambysoft.com/inceptionPhase.html.

Ambler, S.W. & Constantine, L.L. (2000b). *The Unified Process Elaboration Phase*. Gilroy, CA: CMP Books. www.ambysoft.com/elaborationPhase.html.

Ambler, S.W. & Constantine, L.L. (2000c). *The Unified Process Construction Phase*. Gilroy, CA: CMP Books. http://www.ambysoft.com/constructionPhase.html.

Ambler, S.W. & Constantine, L.L. (2002). *The Unified Process Transition and Production Phases*. Gilroy, CA: CMP Books. www.ambysoft.com/transitionProductionPhase.html

Beck, K. (2005). *Extreme Programming Explained 2nd Edition – Embrace Change*. Reading, MA: Addison Wesley Longman, Inc.
邦訳: ケント・ベック, XP エクストリーム・プログラミング入門 : ソフトウェア開発の究極の手法, 永田渉, 飯塚麻理香訳, ピアソン・エデュケーション, 2000

Boehm, B.W. 1988. A Spiral Model of Software Development and Enhancement. *IEEE Computer* 21(5), 61-72.

DeMarco, T. 1997. *The Deadline: A Novel About Project Management*. New York: Dorset House Publishing.

邦訳：トム・デマルコ, デッドライン : ソフト開発を成功に導く 101 の法則, 伊豆原弓訳, 日経 BP 社, 1999

Emam, K. E.; Drouin J.; and Melo, W. 1998. *SPICE: The Theory and Practice of Software Process Improvement and Capability Determination*. Los Alamitos, California: IEEE Computer Society Press.

Graham, I.; Henderson-Sellers, B.; and Younessi, H. 1997. *The OPEN Process Specification*. New York: ACM Press Books.

Graham, I.; Henderson-Sellers, B.; Simons, A., and Younessi, H. 1997. *The OPEN Toolbox of Techniques*. New York: ACM Press Books.

Humphrey, W. S. 1997. *Managing Technical People: Innovation, Teamwork, And The Software Process.* Reading, Massachusetts: Addison-Wesley Longman, Inc.

Jacobson, I., Booch, G., & Rumbaugh, J., (1999). *The Unified Software Development Process*. Reading, MA: Addison Wesley Longman, Inc.
邦訳：イヴァー・ヤコブソン, グラディ・ブーチ, ジェームズ・ランボー, UML による統一ソフトウェア開発プロセス : オブジェクト指向開発方法論, 藤井拓監訳, 翔泳社, 2000

Jacobson, I., Griss, M., Jonsson, P. (1997). *Software Reuse: Architecture, Process, and Organization for Business Success*. New York: ACM Press.
邦訳：I.ヤコブソン, M.グリス, P.ジョンソン, ソフトウェア再利用ガイドブック : アーキテクチャー、プロセス、組織の変革による再利用ビジネス成功への道, 杉本宣男ほか監訳, シイエム・シイ出版部, 2000

Kruchten, P. (2003). *The Rational Unified Process: An Introduction 3/e*. Reading, MA: Addison Wesley Longman, Inc.
邦訳：フィリップ・クルーシュテン, ラショナル統一プロセス入門 (第３版), 藤井拓監訳, ピアソン・エデュケーション, 2004

Copyright 1999-2005 Scott W. Ambler, Ronin International, Inc.
Visit www.enterpriseunifiedprocess.com

Software Engineering Institute. 1995. *The Capability Maturity Model: Guidelines for Improving the Software Process*. Reading Massachusetts: Addison-Wesley Publishing Company, Inc.
邦訳：Carnegie Mellon University Software Engineering Institute,成功するソフトウェア開発 : CMM によるガイドライン, アンダーセンコンサルティング監訳, オーム社, 1998

Vermeulen, A., Ambler, S.W., Bumgardner, G., Metz, E., Misfeldt, T., Shur, J., & Thompson, P. (2000). *The Elements of Java Style*. New York: Cambridge University Press.

Wiegers, K. (1996). *Creating a Software Engineering Culture*. New York: Dorset House Publishing.

Wiegers, K. (1999). *Software Requirements*. Redmond, WA: Microsoft Press.
邦訳：Karl E. Wiegers, ソフトウェア要求 : 顧客が望むシステムとは, 渡部洋子訳, 日経 BP ソフトプレス, 2003

Copyright 1999-2005 Scott W. Ambler, Ronin International, Inc.
Visit www.enterpriseunifiedprocess.com

6. 著者について

Scott W. Ambler は、デンバーに所在するRonin International社のシニアコンサルタントです。Ambler氏は、Prentice Hall PTR から出版されたThe Enterprise Unified Process (2005)、Cambridge University Press から出版された*The Object Primer 3rd Edition* (2004) 及び*The Elements of UML 2.0 Style* (2005)、Wiley Publishing から出版された*Agile Modeling* (2002)及び*Agile Database Techniques* (2003) など多数の書籍を執筆（一部は共著）しています。Ambler氏は、*Software Development* (www.sdmagazine.com)の投稿編集者であり、Enterprise Unified Process (EUP), Agile Model Driven Development (AMDD), Agile Data methodなどの思想的なリーダです。

Copyright 1999-2005 Scott W. Ambler, Ronin International, Inc.
Visit www.enterpriseunifiedprocess.com