運用フェーズにある情報システムに於ける要求一元管理
Inprove requirement management of operation phase

清水勝也
Katsuya Shimizu†
†株式会社エクスライズ
†Exrise Co.,Ltd

要旨
運用フェーズに入った情報システムに於いて，要求管理，システム修正の履歴管理，リリース管理など，複数の管理項目を平行して且つ緊密に連携して実施する必要がある．一方で実際の運用システムでは必ずしも各管理項目が連携されて管理されているとは言い難い．本稿ではモデルケースを通じツールを用いた管理項目の一元管理か，効率化について論ずる．

1. はじめに

情報システムのライフサイクルの中でも運用フェーズは運用信頼性，最終コストの両面から非常に重要であると言える．ところが，実際の開発現場に於いては運用フェーズへの移行に伴い，開発体制の解散，運用チームへの管理移管などに伴ってシステムの保守体制が簡略化される事が一般的である．

簡略化された保守体制に於いてはシステムに対する要求，改修事項の管理，リリース管理などに関して開発フェーズまでとは違った配慮が求められる．

本稿では運用フェーズに於ける情報システムの保守作業に関して，要求などを適切に管理するための一方法論を提案する．

2. 背景

情報システムの運用フェーズに於いては各種の管理情報が多数発生する．運用組織からの要求事項，システムの修正情報，運用作業に於けるインシデント情報などである．

一般にこれらの管理情報は対象の情報システムに関わる組織ごとにばらばらに管理されているのが実情ではないだろうか．これらは本来，統一的に管理される事によって運用上の作業計画や改善提案に生かす事の出来るものである．

システムに関する管理情報が一元管理出来ない原因の一つとして，それらが個別の文書情報として存在しており，容易にデータとして活用できない状態にとどまっている事が挙げられる．そのために複数の情報源からの管理情報を一元管理しようとすると，文章管理の手間が増大してしまい，きちんとした一元管理がされなくなってしまうのである．これには，文献[1]にあるように運用フェーズすなわち保守行程におけるコストがライフサイクル全体の多くの割合を占める為に大きな管理コストのかかる作業を続ける事がコスト上の制約から制限されるためである．

このように，運用フェーズの全体にわたって現実的な管理コストで実現可能な運用管理手法の確率が求められている．

3. 運用フェーズでの管理項目

具体的な運用フェーズでの管理項目としては何があるだろうか.

1. 運用中のソフトウェアに対する要求事項
2. バグ管理情報
3. 運用作業中のインシデント情報
4. 要求事項，バグに対する修正仕様情報

5. ソフトウェアの修正履歴情報
6. ソフトウェアのリリース管理情報

これらの管理情報は情報の管理主体がそれぞれ異なり，またお互いに依存し合う関係にもある．例としてソフトウェアによる要求事項と，それに対する対応実施を行う際の管理情報を挙げる．

1. 要求事項の追加.
2. 要求事項から修正仕様を策定．同時に要求事項の管理状態を修正.
3. 修正仕様に基づきシステム自体の修正実施．修正履歴への反映.
4. 修正を行ったシステムのリリース実施．リリース管理情報の修正．及び要求事項の管理情報を修正.

これらの管理作業を要求事項一つ一つについて行い，また都度，関係するステークホルダーに対する情報提示，相互参照している資料の改訂等が必要となる．管理情報の改訂を行った場合には改訂無いように応じて影響範囲が異なるために，都度影響範囲の評価を行った上で情報管理を行う必要がある．

4. ツールを用いた一元管理

本稿では実際の運用フェーズシステム管理の例として Trac + Subversion を用いた例を紹介する．

4.1. Trac + Subversion

リビジョン管理システムである Subversion とイシュー管理システムである trac の組み合わせで，次のような点を特徴とする．

- wiki 及び「チケット」を用いた情報管理
- 全ての情報について統一した履歴管理を行う
- 各々の管理情報で相互参照が容易に可能
- チケットはタグ付けにより分類管理が可能

4.2. ツールでの管理方法

今回例に挙げる運用システムでは，対象システムを Subversion のリポジトリに格納すると共にシステムに関する管理情報を全て trac の wiki とチケットに入力する方針とした．

作業手順など固定情報は wiki に，要求項目，バグ情報などはチケットに記録する．また，システムを修正し，リポジトリにコミットする際には必ず対応するチケットに対するリンクをコミット時のコメントに埋め込む事をルールとした．

更に，rss 及びメールを用いて管理情報の更新があった時に自動的に連絡がとられるようにした．

4.3. ツールを使った場合の要求事項管理例

Trac + Subversion を用いた場合、3 章で挙げた管理作業がどのようになるか例示する。

1. 要求事項をチケットとして登録する．
2. チケットから未対応のものを抽出し，修正仕様を策定する．作成した仕様ドキュメントをチケットに添付し，チケットステータスを変更する．(ステータスの変更に伴って関係者に自動で通知メールが送られる)
3. チケットから修正待ち状態の物を抽出し，システム修正を行う．完了したら修正版を Subversion リポジトリにコミットし，チケットステータスを変更する．(同様に関係者に自動で通知メールが送られる．また，システムの修正履歴とチケットが連携する事により，自動的に修正履歴となる)

4. 修正を行ったシステムのリリース実施．リリース対象のチケットを抽出し，リリース対象リソースとの確認を行う．

5. ツールによる管理のメリット，デメリット

実際に運用フェーズの管理にツールを適用した事によって以下のようなメリット，デメリットが得られた．

5.1. メリット

- 管理事項がチケットの形で集中管理されるため，課題事項の一覧性が増し，全体把握が容易になった．
- 元になる管理事項のチケットと対応作業の記録がリンクされ，且つ変更情報も記録されるために実施作業の経緯や理由が明確になった．
- 自分の関係する管理情報が変更されたときにメールで通知される上，いつでも自由な条件で抽出表示出来るために実施すべき作業が明確になった．
- システムに対する修正が全てチケットで管理される要求項目と連動しているため，リリース管理が単純になった．
- ツールにより管理事項の未処理，処理済みの区分けが容易になったため、作業漏れが起きにくくなった．

5.2. デメリット

- 関係者を trac 及び Subversion を利用するユーザとして管理する必要があるため，そのための管理業務が発生する．
- trac 及び Subversion を運用するためのサーバが必要となる．
- ツール利用ユーザには自動で周知されるが，ツールを利用しないユーザには情報が通知されない．関係者全員とユーザとして管理するか，ルールを使用しない関係者にも情報提示を行う仕組みやルールの整備が必要．

6. まとめ

手作業で行っていた運用フェーズ情報管理をツール利用する事により運用フェーズでの管理作業の簡略化，抜け漏れの低減を達成する事が出来た．一方でツールによる情報共有がうまくいくにつれ，ツールを使用している関係者と使用しない関係者の間で認識の差異が目立つ局面も出てきた．

運用フェーズでは関係者が複数組織にまたがるなど，関係者の範囲が広くなる事が一般的であるので，全員をツール利用ユーザとして管理する事は現実的でない事が多い．

今後はツールを利用できない関係者を含めて効率的に情報共通が出来るような運用を考慮する必要があると考える．

参考文献

[1] Barry W.Boehm and Philip N.Papaccio, “Understanding and Controlling Software Costs”, IEEETransactions on Software Engineering, Vol.4 No.10, 1988, pp.1462-1477.