保守改修(是正保守)プロセス構築による効果的な欠陥対策とリスク把握に関する取り組み

エクスジェンネットワークス株式会社

井関武史

E-mail：iseki@exgen.co.jp

目次

- 自己紹介
- 背景
- 目標･目的
- 現状分析･課題
- 施策(方針)
- 対策
- 結果
- 結論

自己紹介

- エクスジェンネットワークス株式会社
  - LDAP Manager 部
  - BtoB 向け ID 管理パッケージ製品
    - テスト・品質保証(管理) 担当

- 社外活動
  - Sig-SQA
  - テストの街「葛飾」

背景

背景

- ソフトウェア開発において、どのような予防をしようと欠陥 (バグ) は混入はさけられません。

- 技術・社会(環境) の変化も早く、その変化によってソフトウェア(プロダクト・サービス) の考慮しきれなかった欠陥(バグ) が露呈することも多くなっています。

- 新しいプロダクト・サービスを創造し成長し続けていくことは、価値の提供として重要です。そして、プロダクト・サービスが成長すればするほど、メンテナンスが重要になります。

背景

- 「業務アプリケーション」という特性の場合は、使い捨てではなく、比較的長い保守契約がされるため、その間はメンテナンスし続けなければいけない。
  - 新しい OS、ミドルウェアなどに対応し続けなければいけない

- 新しいアプリケーションを作るときには車輪の再開発とはせずに、派生させて作ることが多い
  - 負債が多いまま、そのまま派生させてしまうと、既存の負債が助長して大きな負債を作ることになる。

背景

- このため、メンテナンスを行うプロセスがないと欠陥 (バグ) の対応が場当たり的な対応となっていき、負債を返却するどころか、負債が増えていくことになります。

- そして、プロダクトに新しい機能などを実装することよりメンテナンスすることに工数がとられることになります。

- 結果、プロダクト・サービスの成長を阻害する事態に陥ります。(余計な負債を抱え込むことになります。)

- そこで、その負債を抱え込んだ事例をもとに、その脱却を行うためのプロセスを定義したこと、その効果について発表させていただきます。

目標・目的

目標・目的

- 本来の品質(価値) を提供する是正改修にしたい
  - 出血を止めるだけのような、場当たり的な欠陥修正にしないこと
- 重要(重大)な課題から解決したい
  - (修正すると) 価値が高いものを選択して修正すること
- 保証する品質を決めてリリースをしたい (テストの深度を決めること)
  - 欠陥の重要性(重大性) がメンバーでばらつきがないこと
  - 「テストしない」理由を答えれること
- プロダクトの成長を阻害させないメンテナンスにしたい (成長の促進)
  - 修正が個別対応にならないこと
  - 間違った修正で、今後の追加、仕様変更ができなくならないようにすること

現状分析・課題

現状分析・課題

- 重要度(重大度) を意識せず修正してリリース (または保留)
  - すぐ修正できるものを選択し修正をして、複雑なものは後回しにされていた
- プロダクトの品質 (背景・価値) を考慮せず「欠陥」の修正という視点で設計
  - 局所的な「欠陥」修正と確認となりデグレが発生
  - 修正した機能で顧客の課題を解決に導けない
  - 修正の意図 (背景) が不明
- テスト設計が CPM 法で保証したい品質が不明
  - ※) CPM 法 とは設計書の語尾だけを「できること」と変えたテスト設計方法
  - (ほぼ) 機能を実証するためのユニットテスト
- 部分的な設計が散在し、どれが正しいか不明
  - 誤った設計書をもとに修正
- 欠陥の測定・分類が困難
  - 保証したい品質のテストがあがっていないのでカバレッジが不明

現状分析・課題

- 重要度(重大度) を意識せず修正してリリース (または保留)
  - 重要 (重大度) が決まっていない総数1300弱の欠陥
  - 修正していくものは、簡単に修正できるもの (ただし、ほぼ価値が出ない)

- 「修正ができるもの」からという
  マインドでの是正改修に
  取り組んでいた

- 処理をキャンセルすると処理をしていない
  ユーザ情報がすべて削除されると
  いう重大な欠陥が残り続ける状態

現状分析・課題

- プロダクトの品質 (背景・価値) を考慮せず「欠陥」の修正という視点で設計
  - 欠陥の直接的な原因(コード)を修正する設計
  - 欠陥によって何が発生している故障、そして事故(アクシデント)が理解できない
  - 修正が部分的になり、少し異なるデータ、環境、設定を利用すると同じような故障が発生
  - 修正に伴うデグレも発生する可能性が大きい

現状分析・課題

- プロダクトの品質 (背景・価値) を考慮せず「欠陥」の修正という視点で設計

【LM7.0】【横展開】LDAP→AD反映 連携先に重複エントリが存在するとDBエラーが発生する

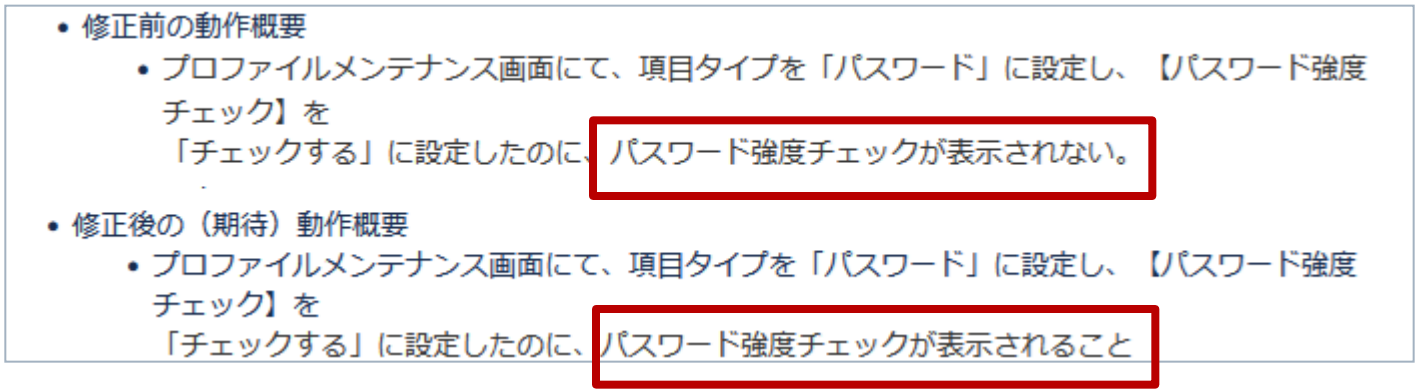
- この「タイトル」だとエラーが出ることが悪いように見える
- 重複エントリーがあった場合、エラーが出ないように「データを削除」すればいい
- という修正しようとしていた。

- そもそも、これはデータ不整合なのでエラーが発生することが正しい
- ※) これは修正を止めました

- プロダクトの本質的な品質を考慮しないと、修正に修正を重ねて、何が正しくなるのかわからなくなる

現状分析・課題

- テスト設計が CPM 法で保証したい品質 (価値) が不明
  - 語尾だけ変わったテスト設計

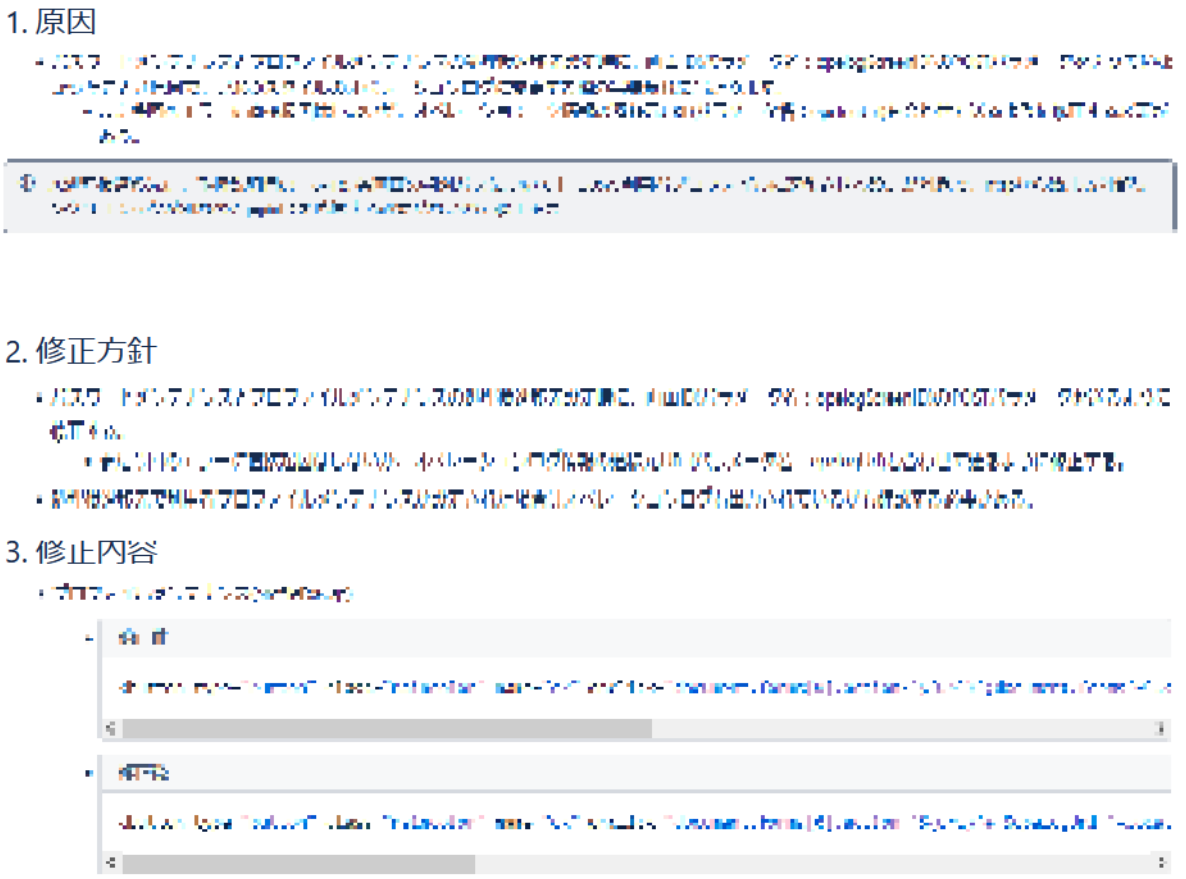


  - 直接的な欠陥は修正・確認はされるが、本来の品質 (価値) が保証できていない
  - 利用者から修正されていないという認識を持たれることが多い
    - 何度も修正してリリースに至る可能性が大きい

  - 本来はパスワード強度の表示によって利用者にパスワード脅威性を気付かせて一定以上の強度のパスワードの設定を促すこと
    - 本来の価値を保証するテストができていない

現状分析・課題

- 部分的な設計が散在し、どれが正しいか不明
  - 欠陥を何度も修正することで以下のような修正することで、二転三転したドキュメントが発生
  - 整理されず次々と修正ドキュメントができ成果物のコードとかけ離れる
  - コードが設計書という状態で、誰も設計 (プロダクトの理念) が把握できない
    - 分からないのでコード見ますという状況



現状分析・課題

- 欠陥の分類・測定が困難
  - 欠陥の修正を繰り返していくことになるため、欠陥修正の欠陥の欠陥修正・・・・という状況になり分類が困難
  - 1つの欠陥を修正するために派生した欠陥修正で測定も困難

施策(方針)

施策(方針)

- 欠陥の重要(重大) 度を決めて、対応する優先度を決定する
- 事故 (アクシデント) を防ぐ修正にする
- どのような品質を保証するテストが必要かを分析・設計をする
- プロダクトの要求・要件 (設計) と是正改修の設計とトレースできる
- 設計書 (ドキュメント) を含めたプロダクトのメンテナンスをする
- 上記の内容を含むプロセスを用意し可観測にする

対策

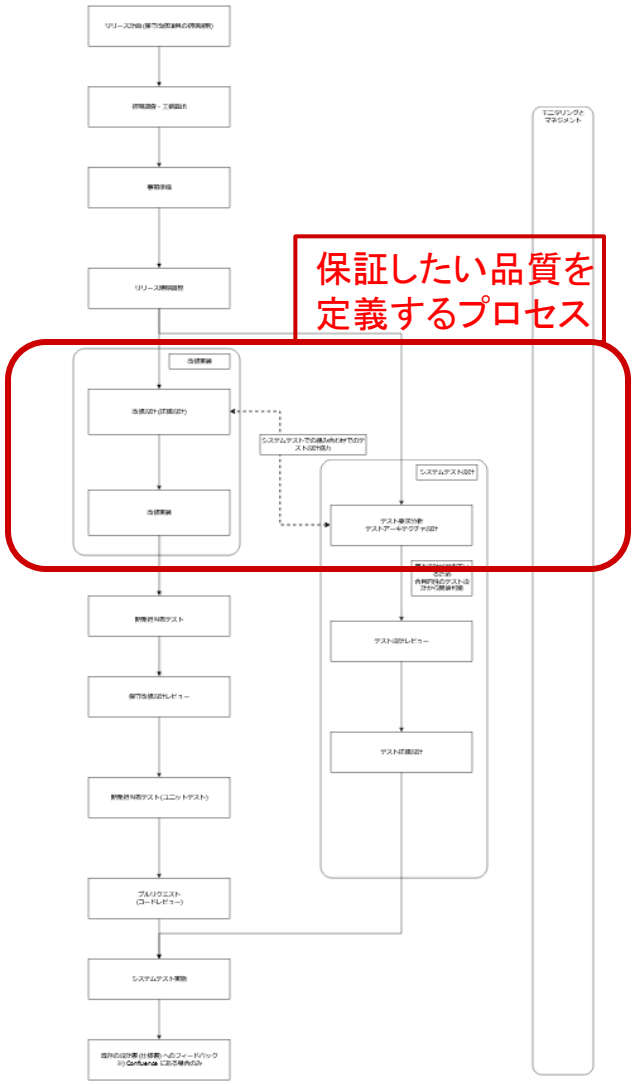
対策

- 是正改修プロセスの定義・構築
  - プロダクトの本来保証したい (すべき)修正方針・内容を検討
  - コードの修正前にテスト設計のリモデリング・リアーキテクチャ
  - 品質保証工程の組み込みとシフトレフト
- 欠陥のランク付け (定性的であるが、ある程度定量的)
  - 欠陥の発生頻度、重要度(重大性)
  - 修正に伴う影響範囲、品質保証レベル
- 欠陥の分類・測定
  - 重要度(重大性)、欠陥対応の工数などを想定
  - ※) 統計的欠陥対応への準備
- プロダクト仕様の成長 (フィードバック)
  - 修正した内容を、本来のドキュメントにフィードバック
  - 従来までの設計書の整備

対策

❑ 是正改修プロセスの定義・構築

- 本来あるべきプロダクトの品質を考慮した改修と、保証する活動として是正改修プロセスを定義
- 保証したい品質を定義するプロセスを上位に入れて、修正方針と保証するテストの内容を確立させる



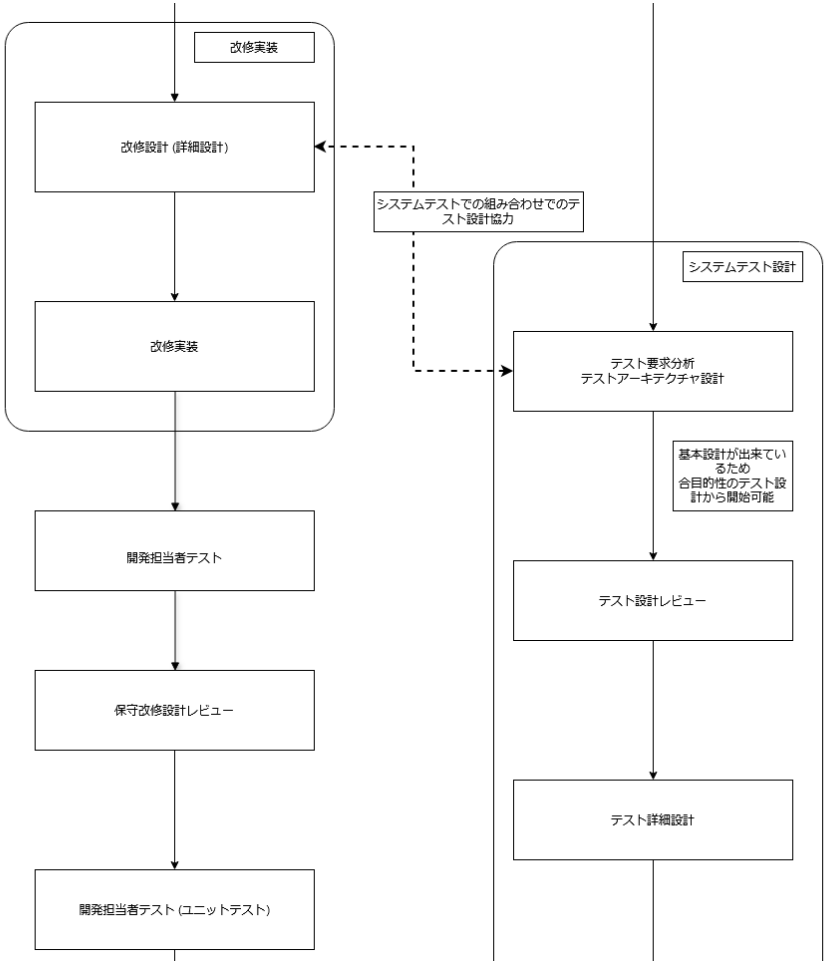
対策

是正改修プロセスの定義・構築

- 欠陥修正の設計開始と同時にプロダクトの分析、すでに存在するドキュメントをもとに設計のリモデリング・リアーキテクチャ

- 以下のことを考慮しテスト設計を
  リモデリング・リアーキテクチャ
  - 保証したい品質(価値)
  - 品質(価値)がどうして必要であるのか
  - どのように品質(価値)を証明するか

- 「欠陥がある」、「実装漏れがある」という状況はモデリング・アーキテクチャで、なにかの失敗しているため、これを発見しないと根本的な修正とならない



対策

❑ 是正改修プロセスの定義・構築

■ 開発担当者

- □ 欠陥対応を行うための設計だけではなく、プロダクト (機能) にどのような品質が必要なのか、なぜ必要なのかを検討します。
- □ そのうえで、テスト方針を決めて実装 (修正) に臨みます。

項目	概要	備考
不具合 (欠陥)・故障	不具合 (欠陥)と、それによってもたらされる故障。	原因ではなく結果による事象を記載する。 そして、その故障によって引き起こされるリスクを分析する。
リスク	故障によって引き起こされる事故を把握する	その事故がどの程度の影響がでるか、どのような条件で発生するか (発生のしやすさもふくめて) 記載する。
期待値	本来のあるべき動作、出力結果	処理の結果だけではなく、どのように (どのような状況で) 動作してほしいのか、なぜこの機能があるのか、その機能でどのようなことが期待されているのかを考慮して期待値を記載する。 ℹ ※) 処理の結果だけだと、コードカバレッジのテストしているのと変わりない。そして、コードカバレッジだけのテストだったらテスト設計なんて必要ない
テスト対象	テストすべきもの (機能、環境、データ)	
保証したい品質特性	リリース時点でどのような品質を保証するのか	バグが修正されている、仕様が変更されているとかは当たり前品質。 自分たちが提供する製品・サービスにおいて何を保証するのかを定義しておくこと。

対策

❑ 是正改修プロセスの定義・構築

■ テスト担当者

- □ 欠陥・故障、リスク(事故)を把握をして、修正によって事故を引き起こすユースケースを把握します。
- □ 修正されたコードのカバレッジを確認(保証)するではなく、事故を防げるかがターゲットです。
- □ 異なる処理などでも同じ故障が発生すると顧客からするとバグが修正されていないと判断される。

項目	概要	備考
不具合 (欠陥) ・故障	不具合 (欠陥)と、それによってもたらされる故障。	原因ではなく結果による事象を記載する。 そして、その故障によって引き起こされるリスクを分析する。
リスク	故障によって引き起こされる事故を把握する	その事故がどの程度の影響がでるか、どのような条件で発生するか (発生のしやすさもふくめて) 記載する。
期待値	本来のあるべき動作、出力結果	処理の結果だけではなく、どのように (どのような状況で) 動作してほしいのか、なぜこの機能があるのか、その機能でどのようなことが期待されているのかを考慮して期待値を記載する。 ※) 処理の結果だけだと、コードカバレッジのテストしているのと変わりない。そして、コードカバレッジだけのテストだったらテスト設計なんて必要ない
テスト対象	テストすべきもの (機能、環境、データ)	
保証したい品質特性	リリース時点でどのような品質を保証するのか	バグが修正されている、仕様が変更されているとかは当たり前品質。 自分たちが提供する製品・サービスにおいて何を保証するのかを定義しておくこと。

対策

- 欠陥のランク付け (定性的であるが、ある程度定量的)
  - 「欠陥」が顧客の(業務プロセス・システム) に及ぼす影響(被害)
  - 修正にともなうプロダクトの影響 (修正範囲・仕様の変更度合い)
  - 上記の2つを考慮して修正すべきか
  - 上記のものを A～D の範囲で確定

  - 上記の定量的に判断することで、修正の優先度を確立
  - 先行して修正(後回しになった修正)なのかを説明できる状態しておく

対策

- 欠陥のランク付け (定性的であるが、ある程度定量的)
  - 「欠陥」が顧客の(業務プロセス・システム) に及ぼす影響(被害)
  - 顧客の(業務プロセス・システム) に対する影響を A～D でレベルつけ
  - それぞれのレベルで、顧客のシステムがどのような状況であるか、そして、修正に伴う設計やレビューの精度を定義

ランク	状況	設計、レビューの担当・精度 保守改修での品質保証・品質管理について (案) | ランクによる品質管理チームによる取り組み 参照	備考
A	顧客の運用が停止する。 製品が停止する。 または、LDAP Manager 以外の IT システムを停止させる等、 顧客環境へ損害をもたらす。	すべての機能において分析を行い、設計書をそろえて開発に臨まなければいけない。(開発チーム同等のすべての設計書が必要) レビューは全員で合意を取るレベルであり、レビューガイドラインにある [チームレビュー] 以上を要する。 プロダクトの保証する品質を明確にしたうえで、リリースしてもよい品質を把握して責任が持てる状態にするため責任者となる PM によるレビューも必要となる。 ※) リリース判定というわけではないが詰び状だすぐらいなので、責任をもった品質になっているかを判断する	全顧客へ修正モジュールの配布を推奨するケース • リリースノートの緊急度が高のレベル
B	顧客の運用に支障があり、回避策がない。 一部機能が使用できない等、著しい制限がかかる。	修正する部分とそれに影響を受ける部分の分析を行い、その部分の設計を行う。 レビューは、すべてのドキュメントにおいて原則全員で合	炎上が見込まれるケース • 先行リリースを要するレベル • リリースノートの緊急度が中のレベル

対策

- 欠陥のランク付け (定性的であるが、ある程度定量的)
  - 修正に伴うプロダクトへの影響度を A～D でレベルつけ
  - それぞれのレベルでどのような状況であるかを定義

ランク	状況
A	機能自体の再作成や、大規模な機能の追加。 製品に対して、全面的に品質を考慮する必要がある。 そのため、基本動作、組み合わせ、状態遷移等、幅広い種類のプロダクト設計・テストを要する。
B	修正対象の機能、または他機能への影響が強い。 修正対象に対して、非機能に関わる品質も考慮する必要がある。 複雑な組み合わせや動作を考慮した、プロダクト設計・テストを要する。
C	修正対象の機能への影響がある。 修正対象に対して、最低限の品質を考慮する必要がある。 部分的な組み合わせや動作を考慮した、プロダクト設計・テストを要する。
D	実質機能への影響がない。 修正対象に対して、品質が保証されている。 ビルドが通れば問題なく、プロダクト設計・テストは不要。

対策

- 欠陥のランク付け (定性的であるが、ある程度定量的)
  - 「顧客の(業務プロセス・システム) に対する影響」と「修正に伴うプロダクトへの影響度」の2つで、保証する「品質レベル」を A～D で判断し修正・テストに臨む

修正に伴う品質レベル

上記の「保守改修 - 改修実装 | 保守改修レベルの総合評価」にある総合評価で、どれぐらいの品質レベルで設計・開発・テスト、レビューを行うかの指針を定義します。

ランク	対処 (品質管理チームが保証すべき品質の精度)
A	様々な製品・機能に影響を及ぼしたり、機能がほぼ全て変更される(新規に作られる)状態であるため、全品質特性を考慮してプロダクト・テスト設計を行う。 全ての設計書の精度は全員で合意を取るレベルであり、レビューガイドラインにある [チームレビュー] 以上を要する。 ℹ ここまでくると、新規開発と同等になるため保守改修として扱えるレベルではないかもしれない。 例えば PostgreSQL 14 に対応したと言いつつも、全然対応できていなかったので全製品の修正・テストのやり直しとか？
B	製品で何をしたいのか(最終的に何ができないといけないのか) を考慮した設計・開発・テストが必要となるもので、必須の品質特性に加え、今回の修正と関連する品質特性も加えたプロダクト・テスト設計を行う。 検討すべき品質特性は、品質特性について に記載されているもの全てである。 (必要がない品質特性も理由付けで排除するため全品質特性を見直す) 全ての設計書は原則全員で合意を取るれベルであり、レビューガイドラインにある [チームレビュー] 以上を要する。 ℹ 主に炎上案件対応となっているもので、局所的にバグが修正されているだけではなく、プロダクト要求、ビジネス要求 (業務フロー) の両方を考慮して対処にあたるレベル。(ただし、特定の製品、機能には限定されている) 製品のオーバーホール的な改修・テストが必要なレベル。
C	局所的な修正で限定された動作となる修正ではあるが影響範囲がいまいち不明だったりする製品や機能にたいして、必須の品質特性のみを考慮して、プロダクト・テスト設計を行う。 検討すべき品質特性は、品質特性について に記載されているもので、[機能性] ([正確性]、[合目的性]、[セキュリティ]) は必須である。 さらに、修正する製品 (機能) に対して 非機能 (機能性以外) の品質でどれが必要かを選択する。 このレベルでは、排除した品質特性は理由付けをしなくてもよい。 (限定される修正であるため) 全ての設計書は、レビューガイドラインにある [ウォークスルー] とする。 ℹ IDワークフローとかによくみられる、同じような処理が複数あってよくわからん状態。 ソースコードを確認してもいいが、テスト分析・設計をして実際に動作させたほうが早いというパターン。

対策

❑ 欠陥の分類・測定

- 欠陥が発生した原因 (要因分析) と修正するレベルを関連付けて、統計的に予防できるようにする。
- ※) 計測・実測中
- 2023年度

		4月	5月	6月	7月	8月	9月	10月	11月	12月	1月	2月	3月	計
工数 (保守改修作業+SetupExe作成依頼作業)		102:35	72:45	67:15	134:05	108:45	76:15	100:20	76:30	32:00	135:50	115:30		
改修規模	A	0	0	0	0	0	0	0	0	0	0	0		0
	B	0	0	0	0	0	0	0	0	0	1	0		1
	C	5	2	16	7	6	4	6	2	4	2	2		56
	D	0	0	0	0	0	0	0	3	0	0	2		5

- 2024年度

		4月	5月	6月
工数 (保守改修作業+SetupExe作成依頼作業)		229:00	120:15	64:00
改修規模	A	0	0	0
	B	0	0	0
	C	14	14	4
	D	0	0	0
	その他	4	4	2
	合計	18	18	6

対策

プロダクト仕様の成長 (フィードバック)

- 欠陥の対応を行ったあと、必ず設計書へフィードバック
- 断片化した仕様・設計にしない
  - 断片化した設計を放置すると混乱を招くことになる
  - どれが最新か、何が真実か、判断できなくなる

- 欠陥を修正したドキュメントは重要ではない
  - 最終的には欠陥修正をしたドキュメントは破棄される

- 本来のプロダクトの設計・仕様のドキュメントにフィードバックして
  正しい(詳しい)ドキュメントにすること

結果

結果

- 欠陥修正に伴う欠陥 (デグレ) は発生していない
  - 保守改修プロセスの導入は2022年度8月から試行
    2023年度から本格的に開始
  - 集計方法・状況などが若干異なりますので、比較が正確ではない

結果

- 増加傾向にあった欠陥 (デグレ) は落ち着いてきた
  - 修正に伴う修正などのチケットが現象
    - 修正に伴う混乱は現象しつつある
  - テスト設計(分析も含む)、テスト実施の工数は1チケットあたり増加しています。
    ※) 品質保証プロセスをいれたため
  - 修正に伴う工数はそこまで変わらない
    - 再修正などの混乱をまねく工数は減少する (2割程度)

	2020	2021	2022	2023
対応欠陥数	184	278	308	64
対応工数	999:30	866:15	1052:45	785.78

- ※) 2022年度7月までの対応欠陥数は欠陥対応での欠陥の対応も含まれている。

結果

❑ 2021年度

		4月	5月	6月	7月	8月	9月	10月	11月	12月	1月	2月
工数 (保守改修作業+SetupExe作成依頼作業)		101.00	135:15	53:00	52:15	96:15	88:45	46:45	90.00	76:30	28:00	88:15
計画・調査		11:45	27:45	0:00	7:45	7:45	10:45	7:00	12:30	5:30	0:00	5:15
開発設計		50:15	66:30	31:45	0:00	54:45	37:15	23:00	42:00	48:00	22:30	53:00
開発製造												
テスト設計		24:30	17:00	5:00	6:30	7:15	8:30	6:45	20:30	9:45	0:15	16:30
テスト実施		8:30	3:15	0:30	3:00	6:00	5:15	4:00	15:00	6:30	0:00	7:00
	合計	19	49	32	17	46	33	11	11	28	28	4

結果

❑ 2022年度

		4月	5月	6月	7月	8月	9月	10月	11月	12月	1月	2月
工数 (保守改修作業+SetupExe作成依頼作業)		194:15	59:30	20:15	101:30	166:15	28:30	101:00	100:30	48:00	28:00	55:45
計画・調査		36:45	18:15	10:30	7:45	7:45	10:45	7:00	12:30	5:30	0:00	5:15
開発設計		96:30	24:30	5:00	58:30	71.00	12:00	43:15	39:15	48:00	24:15	54:30
開発製造												
テスト設計		22:15	5:15	2:00	11:45	24:15	0.00	22:45	41:15	29:00	7:00	19:00
テスト実施		12:00	8:00	2:30	12:45	20:30	1:00	5:00	7:00	23:30	4:00	8:45
	合計	71	8	1	7	42	25	43	11	15	22	49

結果

❑ 2023年度

		4月	5月	6月	7月	8月	9月	10月	11月	12月	1月	2月
工数 (保守改修作業+SetupExe作成依頼作業)		102:35	72:45	67:15	134:05	108:45	76:15	100:20	76:30	32:00	135:50	115:30
計画・調査		10:15	54:05	15:30	8:45	2:45	1:45	5:30	5:15	39:50	17:00	13:30
開発設計				11:15	16:30	5:45	7:15	6:45	8:15	8:00	4:00	14:30
開発製造				7:30	20:15	30:15	16:15	7:30	6:15	27:45	7:15	29:00
テスト設計		21:15	23:30	13:15	49:00	27:00	10:45	28:45	20:45	42:30	9:45	12:45
テスト実施		9:00	16:30	6:30	8:30	5:15	4:00	12:45	15:15	2:15	19:30	0.00
改修規模	A	0	0	0	0	0	0	0	0	0	0	0
	B	0	0	0	0	0	0	0	0	0	1	0
	C	5	2	16	7	6	4	6	2	4	2	2
	D	0	0	0	0	0	0	0	3	0	0	2
	その他	0	0	8(開発チームからのリリース)	9(開発チームからのリリース)	0	0	59(脆弱性対応)	7 (開発チームからのリリース)	0	0	0
	合計	5	2	24	16	6	4	65	12	4	3	4

結果

- 欠陥修正で残存するリスク把握が可能 (事前説明)
  - 全ての品質(価値) を保証する修正は困難
  - リスクがあることを把握したリリース判断
  - 対応できないものであれば仕様・マニュアルに制限事項として記載し周知

#	ステータス	題名	担当者	更新日	優先度	作成者
73238	LM 新規	【LM7.0】アプリケーション名に記号などが入ると設定保存時にマッピングが不正というエラーが発生	LM pending 保留	2024/07/03 11:33	中	井関 武史
73232	LM 新規	【LM7.0】2つ目以降の設定でマッピング関数を利用すると不正なマッピングとなる	LM pending 保留	2024/07/03 10:46	中	井関 武史
72928	LM 新規	【LM6.9】抽出条件外のユーザで「ユーザの削除(停止)処理」になったときに、[削除] マッピング区分の内容で、ユーザ情報が更新されない	LM pending 保留	2024/06/17 17:15	中	井関 武史
72872	LM 新規	【LM6.9】権限委譲したとき、委譲したときのユーザのLDAP属性値を参照している。	LM pending 保留	2024/07/03 10:42	中	井関 武史
72842	LM 新規	【LM6.9】グループ認証で代理承認をしたときの承認者情報(所属組織名)がグループ名で表示される	LM pending 保留	2024/06/13 10:48	中	井関 武史
72227	LM 新規	【インシデント】:パスワードメンテナンスでMac Safari で旧パスワード入力項目を表示すると自動入力で旧パスワードの項目まで設定される	LM pending 保留	2024/05/31 18:11	中	井関 武史
72226	LM 新規	【インシデント】:プロファイルメンテナンスのパスワード入力で keychain で自動入力をした場合に保存されない	LM pending 保留	2024/05/31 18:11	中	井関 武史
72129	LM 新規	【インシデント】: 再パスワード確認項目で、autocomplete の選択画面が表示される	LM pending 保留	2024/05/31 18:11	中	井関 武史
72124	LM 新規	【インシデント】:Firefox で旧パスワード入力項目に自動でパスワード入力される	LM pending 保留	2024/05/31 18:11	中	井関 武史
72017	LM 新規	【LM6.9】PDF出力・CSV出力で「トークンチェックでエラーになりました。不正な画面遷移です。」というエラー	LM pending 保留	2024/05/31 18:11	中	井関 武史

き、委譲したときのユーザのLDAP属性値を参照している。

2024/07/03に更新.

開始日: 2024/06/13
期日:
実開始日:
実終了日:
進捗率: 0%
予定工数:

バージョン	6.9.0.55
環境	-
条件	承認の権限委譲をしたとき
期待値	申請時の承認者の所属属性名 (LDAP属性値) を表示
実施結果	権限委譲したときの所属属性名 (LDAP属性値) が表示

以下に示す権限委譲されたときの承認者情報の所属組織名の値

期待動作とならなかったテストケース・テストパターン

合目的性としてテストをしていた時に発見

2) 機能性 - 機能性以外

テストケース	テストパターン
https://cloud.veriserve.co.jp/projects/2552/test_phases/17981/test_suite_assignments/94983/test_cycles/124233#1	-

※) テストケースを実行するための設定を用意するときに発生。

結論

結論

- 欠陥修正の品質保証を行うプロセスが確立していないと、負債が負債を増加させるため管理工数が増える
- プロダクトの成長期に入るときには、局所的な欠陥対応だけではなく、プロダクトを包括的にメンテナンスを行うプロセスが必要
- 欠陥を修正するときには、本来の品質を考慮した修正をしないと、プロダクトが提供する価値が失われていく
- テストの活動は、欠陥が修正されたかではなく、プロダクトの本来の品質が保証されるかを評価しなければならない