第２分科会

外乱への適切な対応がプロジェクトを救う

Appropriate correspondence to turbulence saves the project

主査：	早川　勲	（株式会社　山武）
副主査：	板倉　稔	（株式会社　イネーブル・ツリー）
研究員：	宮野　洋人	（三菱電機コントロールソフトウェア株式会社）
	菰口　剛至	（三菱電機コントロールソフトウェア株式会社）
	多田　朋之	（株式会社　山武）
	鵜飼　智徳	（株式会社　日立ソリューションズ）

概要

プロジェクトの外部から、現在進行しているプロジェクトとは無関係に問題が降りかかってくる。このような外乱の効果的な対応策を見つけることは容易ではない。

研究員がプロジェクトの現場で遭遇した外乱を分類し、外乱が発生する要因を分析した。その結果、外乱へ組織的に立ち向かうために組織を全体最適化する原理や、システム設計機能とソフトウェア設計機能を峻別して、組織およびルールを設計することなど、効果的な対応策を得た。

本論文では、外乱を分析しそこから導かれる対応策を提案する。

Abstract

The problem happens from the outside of the project without any relation to the project that is progressing now. It is difficult to find an effective countermeasure of such turbulence.

The researcher classified the disorder encountered on the site. And, the cause in which disorder happened was analyzed. As a result, the principle that optimized the entire organization to confront turbulence systematically was obtained. And, it was learnt to distinguish the system design function sharply to the software design function again, and to design the organization and the rule.

In this thesis, it proposes the countermeasure led by having analyzed turbulence.

１．研究の背景と目的

今回の第２分科会には組み込み系の設計開発者からエンタープライズ系の品質管理者まで幅広い人材が集まった。研究員たちは、過去に経験したプロジェクトで対応に苦労した問題や、現在進行しているプロジェクトで対応策が無く困っている問題を抱えていた。そこで我々は、問題の共通点を見出し、対応策を模索することとした。

抱えている問題は大きく 2 つに分けられる。1 つは、プロジェクト内部で発生するプロジェクトで解決すべき問題である。これを内乱と呼ぶ。もう 1 つは、プロジェクト外部から与えられる現在進行しているプロジェクトとは無関係に発生する問題である。これを外乱と呼ぶ。

外乱を解決することは難しく、対応策を容易に見つけることは難しい。我々は外乱を整理し、構造を明確にし、対応策を明らかにすることを目的とした。

２．外乱の事例

本章では、外乱を定義し、外乱の事例を提示する。

２．１ 外乱とは

外乱を次のように定義する。

『プロジェクトの外からの割り込み作業』

この定義を図示すると図 2.1 となる。図中のプロジェクトは、システム開発プロジェクトを想定している。図 2.1 のように、我々が遂行しているプロジェクトには、様々な外乱が飛び込んでくる。例えば、図 2.1 中の教育受講指示や、顧客からの旧製品のクレームなどである。

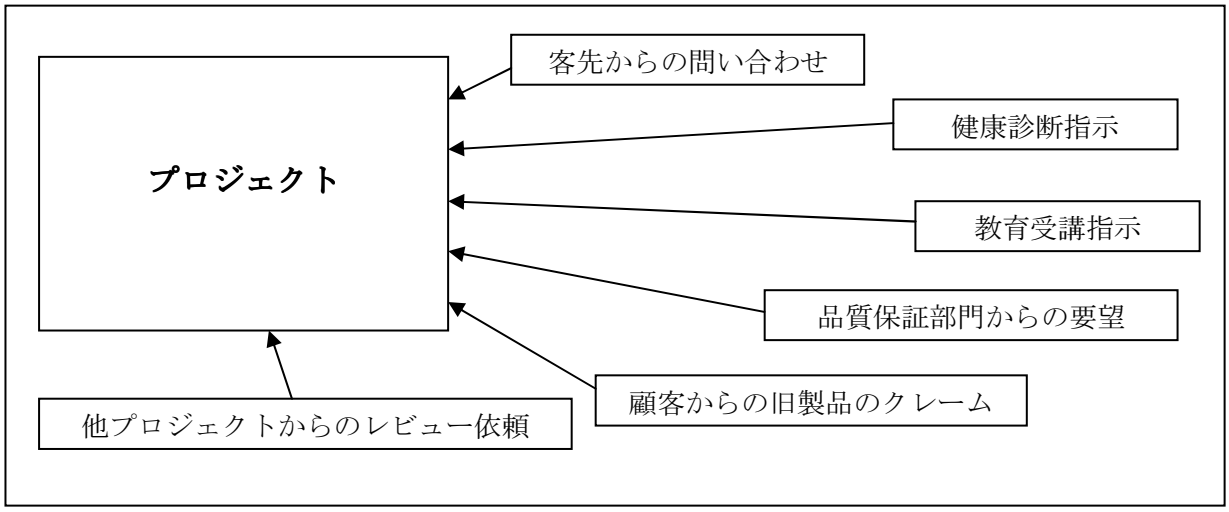

図 2.1　外乱定義の例示

２．２ 外乱の事例

各メンバーから収集した事例を添付資料に示す。調査内容は、プロジェクトにて

①どういった外乱が発生したか？

②どう対処したか？

③対処した結果、遂行中のプロジェクトはどうなったか？

④本来はどう対処するべきであったか？

の 4 点である。

集まった事例は 26 件である。うち 3 件を表 2.2 に例示する。①から③は実データであるが、④は推論である。④は 4 章で言及する。

表 2.2　外乱事例

No.	①どういった外乱が発生したか？	②どう対処したか？	③対処した結果、遂行中のプロジェクトはどうなったか？	④本来はどう対処すべきであったか？
1	旧製品に対してのクレームが発生した	プロジェクト作業を一時停止して、クレームに対応した。	工程見直しや２交代体制により期間短縮し、期間内には完了させることができた。但し、メンバーの負荷が増大した。	余力のあるプロジェクト外メンバーに振る。
2	他プロジェクトからの問い合わせが発生した(以前担当していたプロジェクトのソフトウェアを流用した際の問い合わせ)	プロジェクト作業の合間に対応した。	プロジェクトの合間に対応したことにより負荷が増大した	ソフトウェア仕様書に記述されていれば仕様書を渡すだけで済むはずが、内容が薄いのでソフトウェアの詳細内容が分からない。ソフトウェア仕様書を充実したものにする。
3	旧製品に対しての問合せが発生した（設計・品証部門からのソフトウェアの動作についての問い合わせ）	プロジェクト作業の合間に対応した。	プロジェクトの合間に対応したことにより負荷が増大した	ソフトウェアに機能仕様書に記述されていない処理が多々あるため、ソフトウェアを確認するしか手段がない。記述のないものについては、設計部門に追記するように依頼する。

３．外乱の分類

2 章での 26 件の事例を外乱の内容で分類した結果、表 3.1 の No.1～No.3 に分類できた。No.1 は、メンバーが過去に開発した製品の保守である。No.2 は、今のプロジェクトに直接影響のある組織からの外乱である。No.3 は、プロジェクトとは直接関係の無い人あるいは組織からの外乱である。

その各々を、外乱の発生が予測できるもの「予測可」と、予測できないもの「予測不可」に分類した。結果を表 3.1 に示す。

表 3.1　外乱の分類

No.	外乱の発生内容	予測可	予測不可
1	旧製品に対しての保守	0	1 2
2	品証部門など外部からの条件変更による対応	1	5
3	講座受講、健康診断などプロジェクト作業以外の庶務業務	3	5

表 3.1 から No.1 の「旧製品に対しての保守」で「予測不可」なもの（以降“保守外乱”）が、26 件中の 12 件（46%）で最も多い。そこで今回は、No.1 の「保守外乱」を対象として分析する。

４．外乱の分析

発生した事例に最も適切な対応を取るのに必要なことと、外乱の発生原因を無くすことができないかを以下で分析する。

４．１　保守外乱の対処による分類

保守外乱の対処は、図 4.1.1 ①～④の 4 つに大別できる。

保守外乱が発生した場合は、「自プロジェクトで対処」する場合と、「自プロジェクト以外で対処」する場合と、「外乱に対処しない」場合、の 3 つに分類できる。これらとは別に「外乱を発生させない」ための対処が存在する。

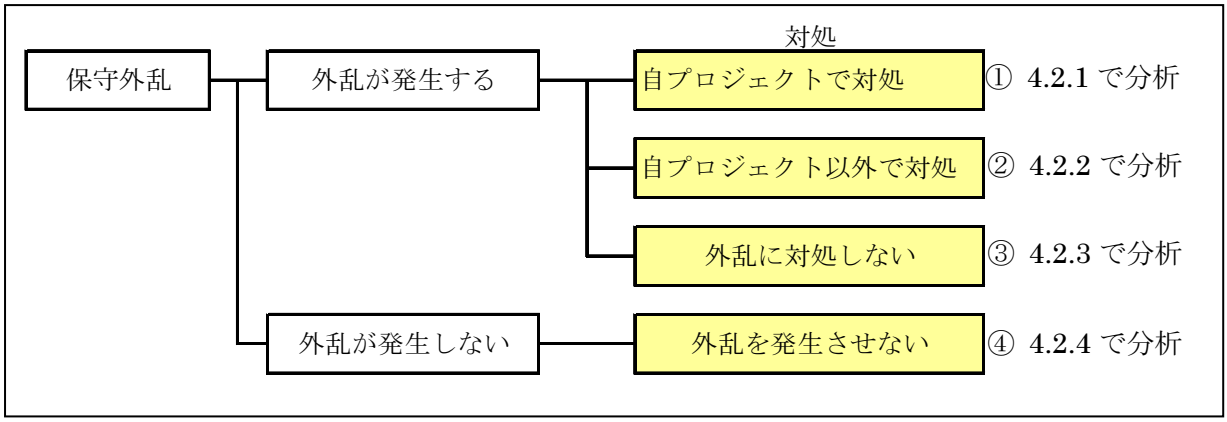

図 4.1.1　保守外乱の対処方法による分類

４．２　保守外乱の対処別分析

表 4.2.1 は、3 章で分類した保守外乱 12 件を抜粋したものである。

表 4.2.1　収集した保守外乱事例

No.	どういった外乱が発生したか	対処した結果どうなったか	実対処方法
1	旧製品のクレーム	工程見直しや２交代体制により期間短縮し、プロジェクトは期間内に完了させた。但し、自分も含めたメンバーの負荷が増大した。	ソフトウェア修正
2	旧製品のクレーム	工程が短縮され、プロジェクト作業を簡略化したものとなった。	ソフトウェア修正
3	現地障害発生（緊急対応）	工程見直し、工期的リスクとして確保していた工数を食い潰した。また試験フェーズでのトラブルが少なく納期には間に合った。	ソフトウェア修正
4	他プロジェクトへのヘルプ	製作フェーズでは 5 日遅れとなったが、試験員のがんばりで試験フェーズで挽回した。	ソフトウェア修正
5	他プロジェクトで問題が発生し、開発要員が引き抜かれる	一時的な作業負荷が増大した。また後工程にて引継ぎミスが発生し品質が劣化した。	ソフトウェア修正
6	保守開発のプロジェクトにおいて、現行システムで障害が発生	一時的な作業負荷が増大した。	ソフトウェア修正
7	他プロジェクトからの問合せ (ソフトウェア設計について)	プロジェクトの合間に対応したことにより負荷が増大した。	問合せ回答
8	他プロジェクトからの問合せ （ソフトウェアの動作について）	プロジェクトの合間に対応したことにより負荷が増大した。	問合せ回答
9	旧製品の問合せ	旧製品の問い合わせについて残業して対応。負荷が増大し、やる気を消失した。	問合せ回答
10	製造から現行製品の出荷検査時に、機能に関して質問を受ける	工程が短縮され、プロジェクト作業を簡略化した。	問合せ回答
11	客先営業サポート （技術的サポートとして客先同行）	工程が短縮され、プロジェクト作業を簡略化した。	客先同行
12	特殊品見積もり （旧製品の特定客先向け機能見積もり）	工程が短縮され、プロジェクト作業を簡略化した。	見積もり作業

表 4.2.1 の事例を、図 4.1.1 の分類（①～④）に従って、以下の(A)から(D)の観点で検討する。

① 自プロジェクトで対処：(A)自プロジェクトへの影響を最小限にする方法がないか。

② 自プロジェクト以外で対処：(B)自プロジェクトへの影響をなくす方法がないか。

③ 外乱に対処しない：(C)外乱に対処しない方法がないか。

④ 外乱を発生させない：(D)元々外乱の発生要因をなくす、あるいは減らす方法がないか。

４．２．１　保守外乱に自プロジェクトで対処

保守外乱に自プロジェクトで対処した事例では、ソフトウェア修正で対処したものが多い（12件中の 6 件で 50%を占める）。多くの場合ソフトウェアが何をどう処理しているかの知識が必要である。しかし、そのような情報は、開発部署の個々人の暗黙知になっていることが多い。このため、担当者以外で調査あるいは対処することが難しい。

問題は、このような外乱を一旦受け入れてしまうとより適切な部署に回すことが難しく、実際には自プロジェクトで対処してしまうことが多いことである。**一旦振られてしまった外乱でも、発行元に無理なく戻せる仕組みと体制を、組織として構築することが望ましい。**

このような仕組みがあれば、一旦外乱をプロジェクトに振られた後でも次のような対策も選択可能になる。これにより外乱の影響の全体最適が図られる。

- 営業部署など現場側の技術力、見積もり力を強化する。
- 保守できるレベルの設計情報、ソフトウェア改修環境の保守資料を整備する。
- 工数を捻出し外注化する。
- プロジェクトの外で対処できる部署、もしくはメンバーを探し移管する。

４．２．２　保守外乱に自プロジェクト以外で対処

表 4.2.1 には、保守外乱であってもソフトウェア修正以外の方法で対処した事例がある。ソフトウェア修正以外の方法で対処したのであれば、自プロジェクト以外（他プロジェクト、他部署、別担当者）で対処できた可能性がある。そこで、本来他で対処すべきであったと考えられる外乱事例を追加収集した。これを表 4.2.2 に示す。

表 4.2.2　本来他部署で実施すべきだった追加事例

No.	外乱種別	詳細	対処内容	根本原因	他に考えられる対処方法
1	旧製品のクレーム	年に一回程度、特定機器が動作停止する。電源再投入しない限り復旧しない。	不具合原因究明の情報収集処理をソフトウェアの改修で追加した。	ハードウェア不良	最初からハードウェア部署と共同で解析する。
2	旧製品のクレーム	月に一回程度、機器故障の警報が発生するが１分程度で復旧する。	ネットワークを分離しトラフィックを軽減した。	システム設計の問題（顧客システム構成毎の設計が必要）	（システム設計部署など）専用の窓口に任せる。
3	旧製品のクレーム	機器の状況表示ランプが実際の状態に合うのが遅い。	性能向上のソフトウェア改修をした。	システム設計の問題（顧客システム構成毎の設計が必要）	（システム設計部署など）専用の窓口に任せる。
4	旧製品のクレーム	数分間動作が停止し自然復旧する。一度だけ発生。	不具合原因究明の情報収集処理をソフトウェアの改修で追加した。	ハードウェア故障	機器故障切り分けのノウハウを現場側で持つ。

（１）表 4.2.2 で、事例 No.1 と No.4 は本来他部署で受けるべきものだが、調査のためには、ソフトウェア開発部門の協力が必要な場合が多い。対応するソフトウェア部門が適切でない場合も有り得るので、4.2.1 章で述べたことと同じく、発行元に無理なく戻せる仕組み、体制が望まれる。

（２）事例 No.2 と No.3 は、本来システム設計の問題であってソフトウェア開発の問題ではない。ソフトウェアは、システムを構成する要素である。システムは、ソフトウェア、ハードウェア、建物など一式が特定目的を達成するよう構成され、動くものである。つまり、ソフトウェア設計（開発）とシステム設計（開発）は本来違う。

カスタム・システムでは、特定の利用者のために開発されるシステムなので、プロジェクトでシステム設計（開発）とソフトウェア設計（開発）が一緒に進められることが多い。標準システムでは、ソフトウェア設計（開発）されたものが、素材として使われ、個々の顧客のシステムになる。個々の顧客のための設計（開発）が、システム設計機能である。ネットワーク化や各製品の融合が進む中で、標準システムのソフトウェア設計（開発）とシステム設計（開発）の境界が不明確となり現場環境に対応できない状況が発生している。

このように、システム開発はソフトウェア設計とシステム設計の両方が含まれる。この両機能をひとつの組織の中に置くか、別の組織に置くか、二つの選択肢がある。

いずれにしても、**システム設計、ソフトウェア設計は別である。従ってシステム設計機能とソフトウェア設計機能を峻別して、組織およびルールを設計すべきである。**

４．２．３　保守外乱に対処しない

表 4.2.1 は保守外乱へ対処した事例だが、保守外乱に対処しなかった事例も収集し分析した。

その結果、どの事例でも客先調査、暫定対策など対症療法をしているので、実際にはプロジェクトへの外乱となっていた。よって「保守外乱に対処しない」という事例がないことが分かった。

また、保守外乱は、当該製品の機能向上のきっかけであることも多い。同様に、外乱に見えるクレームは、サービスや販売方法改善のきっかけになる。従って、組織としては積極的に対応すべきものである。

よって、『保守外乱に対処しない』ことは有り得ないので、検討対象から外す。

４．２．４　保守外乱を発生させない（保守外乱の原因分析）

自プロジェクトでの対処も、自プロジェクト以外での対処も、会社としては外乱を受けている。本来は外乱が発生しないのが理想である。そこで本章では外乱の発生要因を分析することで、元々外乱の発生要因をなくす、あるいは減らすことができないか検討する。

保守外乱の事例を発生要因別に分類したところ、12 件中の 9 件（75%）が仕様問題に起因することが分かった。そこで仕様問題に起因する外乱事例を追加収集し、その要因を分析した。これを表 4.2.3 に示す。

表 4.2.3　保守外乱の仕様問題事例

No.	製品種別	クレームとなった仕様問題	発生原因詳細	原因種別
1	カスタム	管理対象数が不明確だったため、運用開始後に「数」の追加に対応できなかった。	ユーザー側は仕様決定をまだ先で良いと考えてしまい、設計側が見切り発車してしまった。	仕様決定する時期のミスマッチ
2	カスタム	A 操作後の 1 分後に B 操作が可能、8 時間は C 操作不能にする。との仕様であった。結果的に A 操作後すぐに B 操作可能、A 操作後その日は C 操作不能にするだけでよかった。過剰スペックで作成してしまった。	客先の要件をそのまま鵜呑みにしてしまい、背景の認識しようとしなかった。	客先仕様の抽象化ミス
3	標準	ユーザー操作を無視した設計で、ユーザーに著しい負荷を与えてしまった。	ユーザー操作をイメージしなかった。	ユーザー視点欠如
4	標準	担当者にのみメール通知するところ、管理者へもメール送信してしまい、管理者へメールを集中させてしまった	ユーザー側（複数）に立って検証しなかった。	ユーザー視点欠如
5	標準	案内版の表と裏に表示が出るが、表と裏を見るユーザーは違う種別の人のため、表示タイミングを変える必要があった。	ユーザー側（複数）に立って検証しなかった。	ユーザー視点欠如

仕様問題を発生させないために、チェックリストを使ったりレビューを強化したりするが、それだけで仕様問題をなくすことは難しい。その理由は、チェックリストやレビューでは網羅性が保証されないのと、プロジェクト毎の特殊性を反映しづらいためである。

事例のプロジェクトでは、プロジェクトメンバーが実世界を知らないため、使われ方が分からず、利用者の言を鵜呑みにして要求仕様としていた。しかし、利用者と現場を見て利用者と話をした後、要求仕様が出てきた理由が明らかになり、誤解と過剰な仕様を見つけることが出来た。

このことは、プロジェクトメンバーが業務知識を得たからだけではなく、現場に行き、客先やプロジェクトメンバーとの共体験を持ったことにより、自分の仕事に対する責任感と自分の仕事の意味と喜びを体感することが仕事の質を高めたと考えられる。これは動機付けの大きな方法論の一つである。

また、システムが稼働する環境を見て、使用する人から直接ヒアリングすることで、ユーザー視点を持つことができる。ユーザー視点を持つことで、チェックリストの使用やレビューを実施する際に、要件の抜け、誤りなどの気付きが強化される。

５．対策

4 章までで、外乱に対する対策がいくつか明らかになったが、分析の過程として示されているので、どこにあるか分かりにくい。そこで、本章では、対策を再掲し、考察を加える。

５．１　外乱が発生した場合の対策

4.2.3 章で述べた通り、組織は外乱に対処することで得ることが多い。しかし、『実際に対処する組織』と『対処することでメリットを得る組織』が異なることが多い。外乱には対処することを組織の共通理解しておくことにより、実際に対処する組織が外乱を受け入れやすくなる。以下にそのために必要な対策を挙げる。

（１）外乱解決の全体最適化が図れる運営ができること

外乱を一旦受け入れてしまうと、より適切な部署に回すことが難しく、他の適切な部署で対処するべき問題であっても外乱を振られた部署は、疑問を持ちながらも対処してしまうことが多い。より適切な部署で対処されるべきものもあるので、**『一旦振られてしまった外乱でも、発行元に無理なく戻せる仕組みと体制』を構築しておくことが必要である。これにより、外乱の対処が全体で調整され、全体最適に近づくことができる。**

（２）システム設計とソフトウェア設計を峻別して組織とルールを設計すること

ソフトウェアは、システムを構成する要素であり、システムは、ソフトウェア、ハードウェア、建物など一式が特定目的を達成するよう構成され動くものである。即ち、**システム設計とソフトウェア設計は別物である。**

しかし、現実のプロジェクト（機能）の構成では、システム設計（構築）とソフトウェア設計（構築）が分化していないケースが多い。外から見るとシステムもソフトウェアも同じに見えるので、「これはソフトウェア」と簡単に片づけられるケースが多い。そのため**『システム設**

計機能とソフトウェア設計機能を峻別して、組織およびルールを設計する』ことで、組織の機能を明確に表現でき、勘違いや対処部署のミスがへるはずである。

５．２　外乱を発生させない対策

後のプロジェクトで外乱を発生させないために、遂行中のプロジェクトから外乱の要因を除く必要がある。これまでの検討では、外乱の要因の大半は、仕様に関連するものであった。

仕様問題を発生させないためには、要求定義を過不足なく、かつ、必要な精度で定義することである。ところが、開発部隊は、対象業務を知らないことが多いし、実世界の人から要求を聞き出すことも余り訓練されていない。さらに、その前に、開発部隊は、開発に興味をもっているが、利用者に興味をもっているとは言えない。一方で実世界の人は、要求を抽象化することに慣れていない。

そこで、最も根っこにある対象業務に興味を持つことから始めるべきである。それは、現場に行き、客先やプロジェクトメンバーとの共体験を持ち、仕事に対する責任感と自分の仕事の意味と喜びを体感することである。これにより、利用者や利用される状況に興味がわく。これが、要求定義の質を高めることにつながる。

６．まとめ

外乱に適切な対処をすることで製品やサービスを向上させることができる。しかし、そのメリットは外乱に対処する部署が得るとは限らない。そこで、外乱をより適切な部署で扱うことができるようにする対策を得た。

とはいえ、システムの不具合のように、対処してもメリットを生み出さない外乱もある。そこで外乱を発生させない対策も得た。

今回は、外乱の中でも特に多かった「保守外乱」を対象に研究した。今後は、研究結果をプロジェクトで適用することで対応策の内容を充実させるとともに、研究範囲を拡大することでより多くの外乱に対応できるように研究していく。

本論文で提案した外乱への対応策が、読者の抱える問題の解決に役立つことを切に願う。

添付資料

No.	どういった外乱が発生したか？	どう対処したか？	対処した結果、遂行中のプロジェクトはどうなったか？	対処した内容の他に対策・改善・反省点はあるか	外乱の発生源	予測
1	旧製品のクレーム	プロジェクト作業を停止した。	工程見直しや２交代体制により期間短縮し、プロジェクトは期間内に完了させた。但し、自分も含めたメンバーの負荷が増大した。	他の振れる人がいなかった。対処時に、可能な限り新人や他のメンバーと実施し、有識者を増やす。	保守 旧製品のメンテナンス	不可
2	他プロジェクトからの問い合わせ （以前　担当していたプロジェクトのソフトウェアを流用した際の問い合わせ）	プロジェクトには影響はなし	プロジェクトの合間に対応したことにより負荷が増大した	ソフトウェア仕様書の内容が薄いのでソフトウェアの詳細内容が分からない。ソフトウェア仕様書を充実したものにしようにしているが、改善が進まない状況…	保守	不可
3	他プロジェクト問合せ （設計・品証部門からのソフトウェアの動作についての問い合わせ）	プロジェクトには影響はなし	プロジェクトの合間に対応したことにより負荷が増大した	ソフトウェアでは機能仕様書に記述されていない処理が多々ある為、ソフトウェアを確認するしか手段がない。 記述のないものについては、その都度、設計部門に追記するように依頼を行う。	保守	不可
4	研修など業務外の作業	基本的にはプロジェクトには影響はないが、内容によりプロジェクトを停止	プロジェクトは期間内に完了させたが、負荷が増大した。	事前に情報を収集し、計画的に作業を行う	庶務	可 又は 不可
5	現地障害発生（緊急対応）	プロジェクト作業を停止した。	工程見直し、工期的リスクとして確保していた工数を食い潰した。また試験フェーズでのトラブルが少なく納期には間に合った。	最終的に試験フェーズに頼ることになってしまっている。 上位フェーズでの対策が出来ていない	保守	不可
6	他プロジェクトへのヘルプ	プロジェクトメンバーの一人を他工事へヘルプとして派遣。残りのメンバーに機能を分担して遂行	製作フェーズでは５日遅れとなったが、試験員のがんばりで試験フェーズで挽回	BU 内での情報横通しが不足していた為、他プロジェクトで使える要員が余っていたが活用できなかった。現在は、BU 内の横通しを図る目的で定例会を実施している。	保守	可 又は 不可
7	旧製品のクレーム	プロジェクト作業を停止した	工程が短縮され、プロジェクト作業を簡略化したものとなった。	工数捻出し協力会社に依頼する。	保守	不可
8	旧製品の問合せ	プロジェクトには影響はなし	旧製品の問い合わせについて残業して対応。負荷が増大し、やる気消失。	他のメンバーに依頼する	保守	不可
9	客先営業サポート （技術的な質問へ対応できるようサポートとして客先まで同行）	プロジェクト作業を停止した	工程が短縮され、プロジェクト作業を簡略化したものとなった。	断る	保守	不可
10	特殊品見積もり （旧製品の特定客先向け機能変更仕様作成と工数見積もり）	プロジェクト作業を停止した。	工程が短縮され、プロジェクト作業を簡略化したものとなった。	—	保守	不可
11	製造から現行製品の出荷検査時に、機能に関して質問を受ける	プロジェクト作業を停止した。	工程が短縮され、プロジェクト作業を簡略化したものとなった。	—	保守	不可
12	品質ゲート強化により、レビューメトリクス収集依頼（指摘件数、指摘レベル・・）	レビューの為のメトリクス収集を行った	作業負荷が増大した。	前向きに受け入れる姿勢があれば効果があったかも	品質・施策	可
13	品証受け入れ条件がプロジェクト途中で変更となった（残障害は０件にすることなど）	完了させるための理由を記入し、無理やり完了させた	作業負荷が増大した。	—	品質・施策	可
14	品質会議（他製品クレームの状況説明、水平展開会議）	プロジェクト作業を停止した。	工程が短縮され、プロジェクト作業を簡略化したものとなった。	出席しない	品質・施策	不可

添付資料

No.	どういった外乱が発生したか？	どう対処したか？	対処した結果、遂行中のプロジェクトはどうなったか？	対処した内容の他に対策・改善・反省点はあるか	外乱の発生源	予測
15	情報セキュリティ講座受講（急遽開催）	プロジェクト作業を停止した。	工程が短縮され、プロジェクト作業を簡略化したものとなった。	—	庶務	不可
16	作業工番体系の一斉変更 （日報記入上の作業工番が全社統一されたため）	プロジェクト作業を停止した。	残業にて作業を行い、負荷が増大した。	—	庶務	不可
17	協力会社要員の入退構申請 （予想外の入れ替えが多発）	プロジェクト作業を停止した。	残業にて作業を行い、負荷が増大した。	—	庶務	不可
18	健康診断再受信（誤診）	プロジェクト作業を停止した。	残業にて作業を行い、負荷が増大し、体調不良となった。	健康管理、酒を控える	庶務	不可
19	避難訓練	プロジェクト作業を停止した。	工程が短縮され、プロジェクト作業を簡略化したものとなった。	—	庶務	可
20	IT スキル診断	プロジェクト作業を停止した。	作業負荷が増大した。	—	庶務	可
21	社内支援部門（品質保証部など）からの作業依頼。プロジェクトの損益や品質状況の集計のためのデータを提出するよう依頼	プロジェクトには影響はなし 依頼事項は、残業により作業時間を調整し対応した。	作業負荷が増大し、対応者のモチベーションが低下	定期的なもので、かつプロジェクトに関係があるもの（プロジェクトの品質状況調査など）であれば、プロジェクトの作業として計画段階で組み込む。不定期なものや、プロジェクトに関係のないものは、現状打つ手なし。	品質・施策	可 又は 不可
22	社内の集団教育受講 社内で大きな本番障害が発生した場合や、セキュリティ事故が発生した場合に、事象や再発防止策の水平展開が行われる。	プロジェクト作業を停止した。	プロジェクト作業の遅延分を、残業などで調整。プロジェクトメンバーのモチベーション低下	明日はわが身。やむなしとして、受容する。プロジェクト要員に対しては、必要性を理解してもらう。プロジェクトのスケジュール作成の際にある程度のバッファを組み込む。	庶務	不可
23	他プロジェクトで問題が発生し、開発要員が引き抜かれる	プロジェクト要員補充。代わりになる要員が投入できるまでは、残業などで対応した。	一時的な作業負荷が増大した。また後工程にて引継ぎミスが発生し品質が劣化。	開発要員がいなくなることを想定して、情報の共有化をしておく。	保守	不可
24	会社の基準／規格の煩雑化 現場の実態を知らない部署が、新しいプロジェクト管理ルールや、新しい会議体を生み出し、現場に適用を強制する。	プロジェクトにとって意味のあることであれば前向きに取り組むが、大半は時間の無駄であるため、「やったことにする」など大人の対応をする。	規格／基準の形骸化。無駄な規格／基準を生み出した部署は、多くのプロジェクトで適用されたことを理由に、更なる規格／基準を生み出すという悪循環を形成。	とても体力を要するが、無駄な規格／基準に抵抗する。無駄な会議体は中止する。	品質・施策	不可
25	保守開発のプロジェクトにおいて、現行システムで障害が発生	保守開発の内容にも影響を与えるため、速やかに対応（状況によっては２～３日徹夜もありうる）。	一時的な作業負荷が増大した	やむなし。	保守	不可
26	本番障害発生後の後始末 障害報告書の作成、説明、再発防止策検討会議など	熱烈なフォローを受けつつ、愚直に対応。	二度と本番障害を起こさないように品質を確保しようという前向きな考えではなく、本番障害が発生してもいかにして隠蔽するか（もしくは大した障害ではないように見せかける）を考えるようになる。	前向きに反省する。	品質・施策	可 又は 不可