

Requirement Modeling for the C-5 Modernization Program[©]

C-5 M 計画で採用された要件モデリング手法について

Steven D. Allen, Lockheed Martin Aeronautics Company

Mark B. Hall, Lockheed Martin Aeronautics Company

Mark D. Mansfield, Lockheed Martin Aeronautics Company

Verlin Kelly, Lockheed Martin Aeronautics Company

Dr. Mark R. Blackburn, Systems and Software Consortium

この資料では、Lockheed Martin Aeronautics Company C-5 Modernization (C-5M) 計画における航空機エンジンの性能向上、信頼性向上のために取組まれた、要件ベース・モデリング (*a requirement-based modeling*) の成果について紹介する。要件ベース・モデリングの成果として、より一貫性があり、完全で、正確な要件とインターフェイスの情報から、デザインや実装のプロセスが支援された。システムエンジニアは、要件モデルの正当性を確認するためにシミュレーションを用い、相当数の要件上の欠陥を検出し、ソフトウェア実装前にそれらを修正することができた。

ロッキードマーチン社の航空関連事業 (Lockheed Martin Aeronautics Company) の C-5M Program は、航空機の全体的な信頼性を向上させる、航空機エンジンとそれに関わるソフトウェアシステムの改善である。何名かの C-5M プロジェクト要員は、C-130J Avionics System の要求仕様を開発する際に、SCR (the software cost reduction) 要件モデリング手法を用いることに成功していた。[1] 要件ベース・モデリングでは、正確な振舞いの要件の作成、インターフェイス情報の形式化 (formalizes interface information) が、開発ライフサイクルの早期段階で行えるようになり、デザイン・実装のプロセスを支援する。C-5M プログラムには、航空機ソフトウェアの仕様、デザイン、実装工程を向上させる様々な関連する戦略が実施された。これらの戦略により、主なる機能ブロックの、リリース後の問題報告や修正を飛躍的に削減しながら、スケジュールどおりのリリースをサポートできた。この資料では、要件モデリングと早期の要件バリデーションを行うことで、要求の仕様化プロセスが向上することについてフォーカスする。

要件モデルは、コンポーネントへのインターフェイスの観点から定義される、コンポーネントの要求される機能の振舞いをフォーマルに仕様記述したもの (形式的な仕様)。モデリングのプロセスで、モデリングツールによる自動解析により、要件やインターフェイスの問題検出がサポートされる。加えて、システムエンジニアは要件のシミュレータを用い、モデル化された要件の正しさを事前に (デザインや実装前に) 確認できる。要件シミュレータは、要件モデルを取込み、GUI を用いてモデル化された機能の振舞いに対するシナリオの実行をサポートする。シナリオは、一連の入力イベントと、その結果として対応する内部ステートと出力の変化。シナリオは、ハイレベルなシステムのユースケース、あるいはローレベルなコンポーネントの相互作用。シナリオのシミュレーション時に観測される予期されないステートや出力は、多くの場合要件の欠陥である。

モデリングプロセスと要件シミュレーションにより、多くの要件上の欠陥が発見された。幸いにもこれらの欠陥は、プロジェクトの早期段階で特定され、ソフトウェア実装プロセスの前に訂正された。IPR (Integration problem reports 統合テスト時に明らかになる、実装、デザイン、要件間の欠陥 (矛盾、一貫性の無さなど) のレポート) は、要件、デザイン、実装の欠陥に対する主要な尺度となった。多くの IPR は、以前の同様のプロジェクト (C-5 Avionics Modernization Program (AMP)) と比較して大幅に削減された。この資料では、成果として主張するプロセスの向上と開発計画で得られた恩恵を、証明する IPR の尺度とその分析結果について紹介する。このデータは、C-5AMP と比較して、C-5 計画のプロセスでは早期に欠陥を発見し、欠陥が約半分になり、平均して 2 倍早くに修正されたことを立証する。

C-5M 計画は、今後数年に渡って新しい開発リリースを継続する。継続中の C-5 AMP には、C-5M 計画で成功裏に適応された改善プロセスの適応が計画されている。C-5AMP で実施された調査により、大規模・複雑なアップグレードの開発、保守は、この資料で紹介する改善プロセスを用いることで、よりコスト効率良く行えることを示唆する。

プロセスの概要

C-5Mプロジェクトは、離れた地域間で構成される統合開発チームで実施されている。以下の図 (Figure 1) では、ターゲットソフトウェアに帰結するまでの、各役割と成果物の流れの概要を示す。そしてC-5Mソフトウェアの開発で採用された、トラディショナルなプロセスの段階と役割が図の上側、モデリングによる拡張が図の下側に示されている。システムエンジニアはテキストによる要件を開発し、SRS (a software requirement specification) に取込まれる何らかのタイプの分析モデルを同様に開発する。SRS要件は、構造化されたシンタックスと限られた動詞 (e.g., acquire, validate, provide, and derive) を持つRSL (a requirement specification language) を用いて開発される。RSLは、SCRモデリング手法を補足するために開発されたものである。同時に、連続的な要件のフローダウンのプロセスがある。主導的立場にいるソフトウェア・アーキテクト (lead software architect) はソフトウェア・アーキテクチャのコンポーネントを識別し、ソフトウェア要件モデリング担当者 (software requirements modelers) と協調して、要件と関連するインターフェイスをモデルにフォーマル化 (形式化) する。

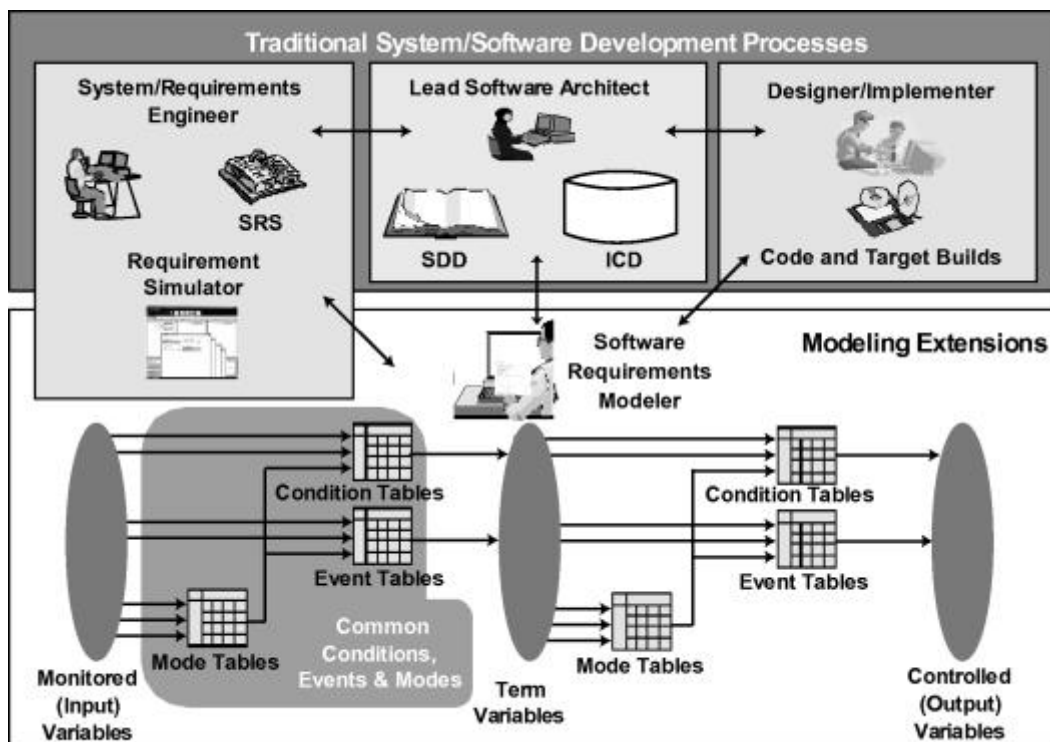


Figure 1: Process Roles and Flow

ソフトウェア要件モデラー (software requirements modeler) は、SCR 手法をサポートするモデリングツールを用いて、SRS と ICD (interface control document) から要件モデルを開発する。モデルは、振舞いの要求と、ICD から引き出されるインターフェイスの情報 (e.g., inputs, outputs, types, and ranges) を取込む。モデリングのプロセスにより、システムエンジニアとソフトウェア・アーキテクトの相互間の活動を介して解決されるべき、要件あるいはインターフェイスの問題が、確認される。

例えば、インターフェイスの様子は、サブコントラクターを含むプロジェクトチームで共有されるデータベース内に取り込まれていた。しかしながら計画の早期段階では、それらは通常完全ではなく、あるいは一貫したものでなかった (例えば最初の 100 日間)。要件モデリングのプロセスと、それに用いられるツールにより、インターフェイスの情報は完全で、一貫したものになる。更なる追加の問題、異常はモデルによる要件シミュレーションを介して、システムエンジニアによって特定される。正当性が確認された要件モデルは SDD (the software design document) にリンクされる。デザイナー、実装担当者は直接 SDD、要件モデル、コード実装の為にインターフェイスから作業を行う。プロセスに対するこれらモデリングに関連する拡張は、チームの総合的能力向上に貢献する。より良い要件、インターフェイスのドキュメントにより、ソフトウェアデザイナーは、要件上の問題を掘り下げることや、仮定を思い巡らせるような費用負荷の大きいやり直し作業をするかわりに、詳細設計、コード実装に集中できるようになる。

インターフェイス・ドリブンな要件モデリング

SCR は表形式のモデリングツールで、多くのエンジニアにとって学習しやすく、効率的なツールである。SCR モデリング言語は、明確に定義されたシンタックス、セマンティックを持ち、正確で、ツールによる解析が可能な、振舞いの要求の仕様化をサポートする。モデルは、コンポーネントの機能要求を、モニターされる変数 (monitored variables (inputs)) とコントロールされる変数 (controlled variables (outputs)) に関連したテーブルを用いて表現する (Figure 1 を参照)。これには x 3 つの基本タイプのテーブルがある。これらは、1) mode transition tables, 2) event tables, and 3) condition tables である。“mode transition tables” はステートマシンで、ここで関連するシステムのステートはシステムモード (system modes) と呼ばれ、ステートマシンの遷移はイベントにより決定される。“event” は、なんらかのシステムの実体が値を変更すると発生する。“condition” は、システムの状態 (state) を決定する述部となる。“term” は入力変数、モード、あるいは他の term を置き換えて定義されるファンクション。SCR テーブルは、モニターされる変数 (monitored variables (inputs)) とコントロールされる変数 (controlled variables (outputs)) 間の複雑な関係を定義する為に、モード、あるいは terms 変数を用いて組合わされる。そして、共通のコンディション、イベント、モードが一度定義されると、何回でも参照される。

モデル開発者はゴール指向アプローチを取り、以下に示す全体的なガイドラインに沿って機能とコンストレインツ (制約) をコンポーネントの各出力 (controlled variable or term) に対して特定する :

- 各計算された値を表の出力とする表を作成する。値は単純な割り当て、あるいは複雑なファンクションで指定できる。これは、RSL のアクション動詞の “provide” と対応する。
- コンディションテーブルを用い、出力 (あるいは term) への関係を記述する。関係が一定時間継続する場合。全コンディションが、各出力の割り当てで TRUE であることを特定する。コンディションは、RSL 動詞の “validate” に関連する。なぜならこれらは、入力に対するコンストレインツであるため。
- イベントテーブルを用い、出力 (あるいは term) への関係を記述する。関係が特定の時点で定義される場合。各出力を割り当てるイベントをトリガーする、イベントとオプションでガードコンディションを定義する。
- モードトランジションテーブルを用い、入出力 (あるいは term) などオブジェクト間の関係を記述する。モードへの関係が一定の期間、定義される場合 (一連のシステム状態)。一連のモードを特定し、モードトランジションテーブルの各ソースからデスティネーションへの遷移に結び付くイベントを定義する。
- 2 つ以上の出力 (or terms) のファンクションのコンストレインツ (i.e., conditions or events) に関連する共通のコンディションがあると、他のテーブルから参照されうる “term” テーブルを定義する。“term” リファレンスは、テーブルアサインメント、コンディション、イベントとして機能する。“term” 変数は、RSL 動詞の “derive” に相当する。“term” は他のテーブルから参照される中間値であるので。

インターフェイスドリブンなモデリングアプローチの詳細は、[3]を参照。

要件のバリデーション

モデリングプロセスにより顧客要件は、エンジニアに理解される言語に変換される。これによるフォーマル化(形式化)は、システムからソフトウェアインターフェイス境界まで至って、システム要件エンジニア、ソフトウェア・アーキテクト、要件モデラーの共通の理解を確実にする。にもかかわらず、テキストベースの要件開発担当であるシステムエンジニアは、当初要件シミュレータ(SCR手法をサポートした)を使用することに気が進まなかった。しかしながら、幾分のシナリオを開発したあと、彼らの要件に対する理解が、システムの複雑さゆえに、不完全、あるいは不正確になりうることを理解した。彼らはドメインの知識を持ちモデル化された要件の正しさを判断できるため、彼らのインプットをバリデーションプロセスに加えることは大変重要であるため、この改訂された観点は、欠く事のできないものであった。シナリオのバリデーションの標準的なプロセスは、GUIシミュレータを用いイベントのシーケンスに相対する入力値をシステムエンジニアが設定することを必要とする。各入力のと、システムエンジニアはシステムの出力と内部状態(termなど)を観測し、その正当性を確認する。モデル上の問題は、イベントのシーケンスが特定シナリオで期待する出力と異なる結果となった時に露呈される。要件のバリデーションを通して検出される最も一般的なタイプの問題は、関連するモデルのコンディション、モード、イベントテーブルなどの矛盾から得られる。同じ矛盾はテキストベースの要求仕様にも存在するが、SRSドキュメント内で何百ページにもおよぶ要件のそのような矛盾をマニュアルインスペクションやレビューで検出することは困難である。要件モデルでは関連する要件間をフォーマルにリンクし(Figure 1)、シミュレーションや自動解析などのツールを用いて矛盾や問題(e.g., logical contradictions, potential divide-by-zero situation)を検出することを支援する。

測定データ

C-5Mアビオニクスソフトウェアは、リリースされる各ソフトウェアのブロックを積み重ねるようにして増築式に開発された(各ソフトウェアブロックのリリースはスケジュールどおりで、完全な機能を提供)。これら完全な機能で、オンタイムにソフトウェアをリリースしたことは、改善されたプロセスの有効性を証明するエビデンスとなった。しかしながら、C-5M計画、C-5AMPで調査されたIPR(Integration problem reports)に関する測定データは、客観的にこれらプロセスの比較情報を提供する。理想的には各計画は、1) IPRの総数を最小化する、2) IPRに関連する欠陥を早期に検出、3) IPRに関連する欠陥を修正する期間を短縮する、といった目標をもつべきである。以下のIPR測定データは、明白に規定された目標に対する比較情報となる。4つの基本尺度と1つの派生尺度が、C-5M計画のプロセス向上の実証に使用される。

- 基本尺度：最初のIPRからの日数
- 基本尺度：IPRの数
- 基本尺度：IPRがオープンされた日
- 基本尺度：IPRがクローズされた日
- 派生尺度：IPRがクローズされるまでの平均日数

C-5 測定の分析担当は、C-5M の IPR は C-5AMP の IPR と比較して、より詳細で具体的であったと証言している。より詳細で正確な要件が C-5M では定義され、改善されたプロセスを用いたため。例えば、C-5M の IPR は以下のように、具体的な内容が記載される：

“ 圧力が限界地を超える時、圧力計の表示は赤であるべきで、黄色ではない ”

同様の IPR が、C-5AMP では、

“ ウォーニングメッセージに、正しくない色設定がある,,, ”

結果、C-5AMP の IPR は、C-5M の IPR と比べて内容が曖昧なため、調査し、必要となる修正作業・実装をするためにより多くの時間が必要で、2 倍以上の工数が必要となった。

次のセクションでは、この主張をサポートする尺度のデータを紹介する。例えば、C-5AMP の IPR が是認されるまでの日数は、時に C-5M 計画のそれの、およそ 3 倍となった。ただし、IPR の総数と、IPR の是認までの日数は、計画の守秘上公開していない。

尺度の分析

Figures 2, 3, 4 では、C-5M のデータを C-5AMP のものに対して、各計画のスタートから 100 日の区切りごとで比較している。Figures 2 では、IPR の累積総数を表している。最初の 400 日間は、C-5M 計画の IPR 数は、C-5AMP のものよりも僅かに多い。C-5AMP の IPR 数は、計画の約 500 日で著しく増加している。800 日では、C-5AMP の IPR 数は、C-5M の 2 倍となっている。Figure 2 では、C-5M 計画の今後 200 日間の IPR 数を線形予測した数量を記載しているが、その予想では C-5AMP の IPR 数が 3 倍となる。この予測は Figure 3 の測定尺度により実証される。

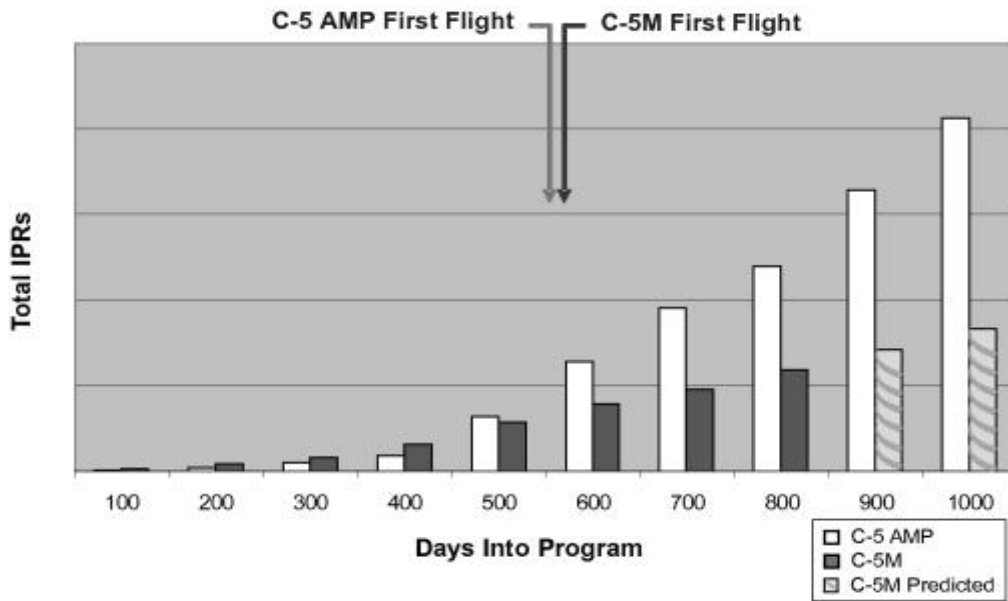


Figure 2: IPRs vs. Days Into Program 100日ごとの、IPRの総数

Figure 3 では、100 日間ごとで追加された IPR 数を表示している。ここで、C-5M 計画のほうが、早期に多くの IPR が検出されたことがわかる。これは、要件モデリングとバリデーションにより、早期の欠陥の検出が支援されることを示唆する。Figure 3 では、500 日目からは、C-5AMP の IPR が著しく増加したこともわかる。C-5AMP の IPR は、その時点で C-5M のおおよそ倍であった。C-5M の IPR 発見の割合は、C-5AMP の 500 から 800 日目までの割合に比べて、相当低くなっている。

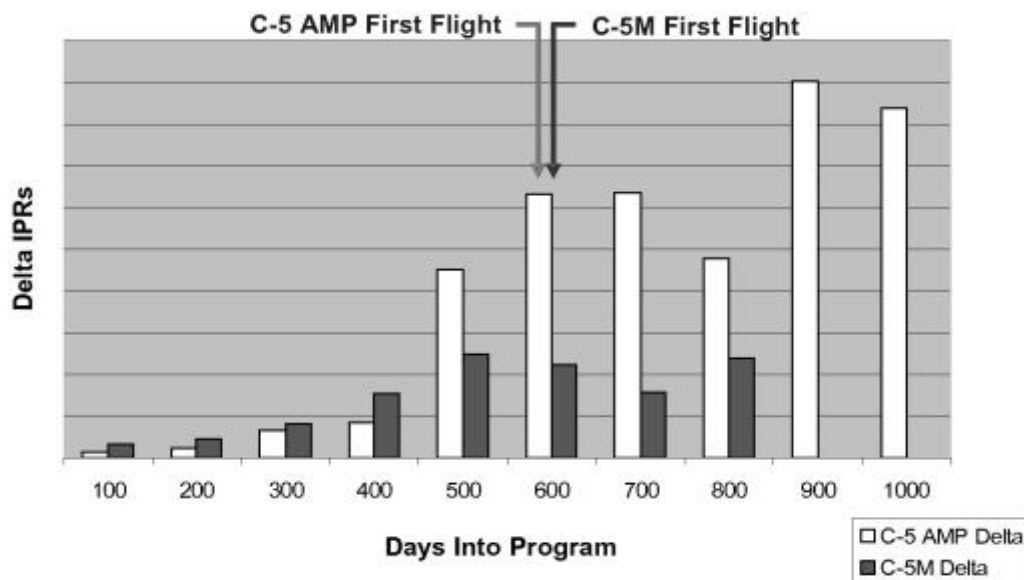


Figure 3: Delta IPRs vs. Days Into Program 100日ごとで追加された IPR 数

Figure 4 は、IPRに関連した欠陥を修正するシステムの変更を是認するのに要した日数。C-5M計画のIPR是認プロセスは、始めの段階では長くかかった。この資料で定義したプロセスは、C-5M計画の最初の 100 日間は完全に洗練されておらず、改善作業が計画実行当初に取り計らわれた。しかしながら、是認期間の平均は 800 日まで連続して削減されたことをデータは示している。これらのデータを合わせて考えるとC-5AMPと比較して、800 日目で、C-5M計画で検出されたIPR数は少なく、しかし欠陥を修正するのに要した時間は著しく速かったことがわかる。これらデータを複合的にみること、要件モデリングとそのバリデーションは、早期欠陥検出を飛躍的に改善し、迅速に欠陥を排除し、修正できるという結論が導かれる。

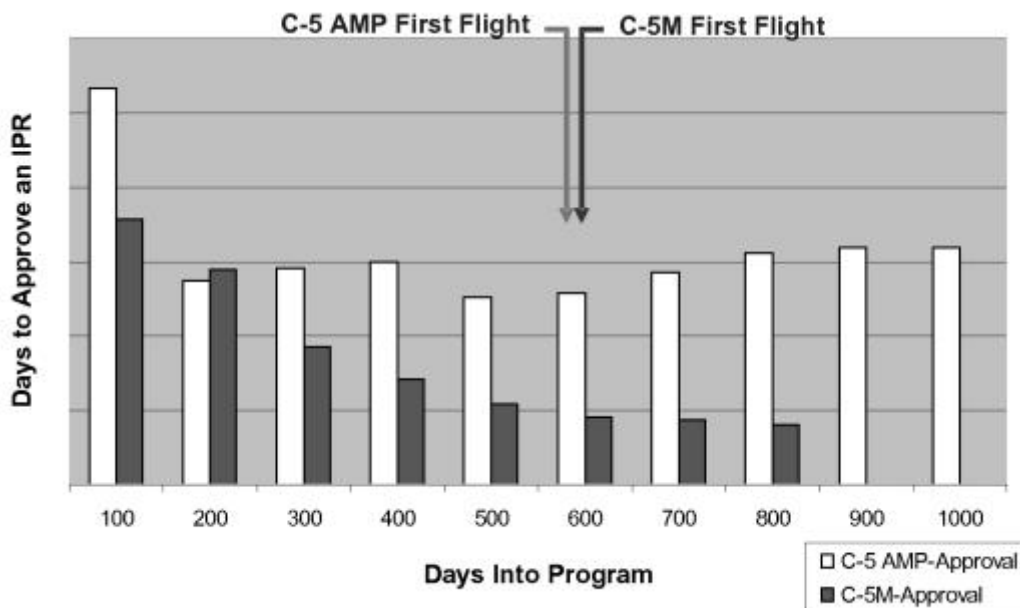


Figure 4: Average Days to IPR Approval IPRの是認までに要する平均日数

モデルの活用

おおよそ 90%ものソフトウェア詳細設計記述は、要件モデルを基にする。要件モデルは、テキストで記載されたデザインの代わりに、SDD (the software design document) にリンクされる。モデルは、ハイレベル、ローレベル (i.e., derived requirements 派生要求) の両方の要件をカバーする。一般的ではない、あるいは複雑なデザインは SDD にテキスト、フローダイアグラム、その他の描画ツールなどを用いてドキュメント化される。これは、要件モデリングプロセスを活用することで得られる、別のプロセス向上効果である。モデルは、SDD 内で直接参照される要件の、フォーマル (形式的) で正確な記述を提供した。

データ検索、データバリデーション、フィルタリングなどの共通のモデリングパターンは、見出され、異なるソフトウェアコンポーネントに展開された。システムエンジニアとソフトウェアモデラーの先導者は、パターンをバリデートし、モデルテンプレートとして利用できるようにした。モデルテンプレートにより、モデリングプロセスのさらなる一貫性の普及に貢献した。

まとめ

この資料では、C-5M 計画に要件モデリングを採用することで得られた成果とアプローチについて紹介した。要件モデリングは、デザインや実装プロセスをサポートする、より良い要件とインターフェイス情報の開発を支援する。システムエンジニアは要件モデルのシミュレータを用い、要件の正確さと一貫性を確認した。

要件モデリングとシミュレーションプロセスにより、相当数の要件上の欠陥を、ソフトウェア実装前に検出できた。加えて、測定された尺度はプロセスの向上と開発計画にもたらされた利益を立証した。測定された尺度により、C-5AMP 計画に比べて、C-5M 計画のプロセスでは欠陥を早期に見出し、欠陥総数は半分で、欠陥是正に要する期間が平均 2 倍速かったことを確認できた。また、システムとソフトウェア要求の早期バリデーションにより、Lockheed Martin Aeronautics Company と、その顧客に相当な利益がもたらされたことも公表されている。C-5M 計画でもたらされたこの大きな成果は、要件モデリングプロセスを C-5AMP 持続の取り組みとしての大規模・複雑なソフトウェアシステムのアップグレードにも適応させる計画が決まった。

References

1. Faulk, Stuart, et al. *Experience Applying the CoRE Method to the Lockheed C130J*. Proc. of the Ninth Annual Conference on Computer Assurance, IEEE 94CH34157. Gaithersburg, MD, June 1994: 3-8.
2. Kelly, V., et al. *Requirements Testability and Test Automation*. Proc. of the Lockheed Martin Joint Symposium. June 2001.
3. Blackburn, Mark R., Robert D. Busser, and Aaron M. Nauman. "Interface-Driven, Model-Based Test Automation." May 2003 <www.stsc.hill.af.mil/crosstalk/2003/05/blackburn.html>.

Note

1. At the request of Lockheed Martin management, the horizontal line values for Figures 2, 3, and 4 have been removed.

About the Authors

Steven D. Allen is a staff engineer with the Lockheed Martin Aeronautics Company. He has more than 24 years of experience in the analysis, design, and development of software for integrated software subsystems. In his role over the last few years as a requirements engineer for the mission processing subsystem of the C-5M Program, Allen has been active in the process, procedures, and use of current software tools to clearly define and validate the software system requirements through the use of requirement modeling, simulation, and verification techniques.

Mark B. Hall is a senior staff engineer with the Lockheed Martin Aeronautics Company. He has more than 20 years of experience in the analysis, design, and development of software for integrated avionics. As the software architect for the mission processing subsystem of the C-5M Program, Hall has been active in process and procedures to clearly define the software requirements and validate them through the use of modeling and simulation. He has a bachelor's degree electrical engineering from Southern Polytechnic State University and an MBA from Kennesaw State University.

Mark D. Mansfield is a staff engineer with the Lockheed Martin Aeronautics Company. He has more than 10 years of experience in the analysis, design, and development of software for integrated avionics. In his role over the last few years as the systems engineer for the mission processing subsystem of the C-5M Program, Mansfield has been active in the process and procedures to clearly define the system and software requirements and to validate them through the use of modeling and simulation. He has a bachelor's degree in aeronautics from Embry-Riddle Aeronautical University.

Verlin Kelly is a staff specialist for the Lockheed Martin Aeronautics Company and has 41 years of experience in embedded systems and software and automated support systems. He has developed software engineering training curriculum and courses as well as co-authored presentations to the Systems and Software Technology Conference on "Quantitatively Managing Multi-Company Software Teams" and "Requirement Testability and Test Automation." Kelly has a master's degree in operational mathematics from the University of Texas at Arlington (UTA) and a bachelor's degree in physics and mathematics from Baylor University. He is involved in system/software Unified Modeling Language development and automated testing and has served on the industry advisory council for the computer science and engineering departments at the UTA and Texas Christian University.

Mark R. Blackburn, Ph.D., is a Systems and Software Consortium fellow and co-inventor of the T-VEC (also known as test-vector) system. He has more than 20 years of software systems engineering experience, spending most of his time helping companies adopt model-driven engineering tools and methods. Blackburn is a frequent speaker at conferences and symposia, and has authored more than 70 papers covering a broad spectrum of topics such as modeling, verification, software safety, security, reliability, and measurement. He has a bachelor's degree in mathematics from Arizona State University, a master's degree in mathematics from Florida Atlantic University, and a doctorate in information technology from George Mason University.

Systems and Software Consortium
2214 Rock Hill RD
Herndon, VA 20170
Phone: (703) 742-7136
Fax: (703) 742-7350
E-mail: blackburn@software.org

© 2009 Systems and Software Consortium, NFP. All rights reserved.



富士設備工業株式会社 電子機器事業部

<http://www.fuji-setsu.co.jp>

〒591-8025 大阪府堺市北区長曾根町1928-1 Tel: 072-252-2128