車載システム製品へのコンカレント開発の適用

－新規開発における開発期間短縮へのアプローチ－

Concurrent Development applied to Software Development of Automotive System

- Approach to Improving the Lead Time for New Development -

株式会社デンソー　走行安全技術４部
DENSO CORPORATION Driving Assist & Safety Eng. Div.4
○林　健吾
○Kengo Hayashi

Abstract In this paper, we propose the approach to concurrent development in software development of automotive system. The software in the system is increasing its size and complexity and required to be developed in shorter lead time. We developed the software in the two phases of “prototype development for developing requirement significations” and “product software development based on the requirement specifications”. We propose development of requirement specifications using prototype software in the product software development as an approach to the concurrent development. We demonstrated it in our automotive system development and improved almost 50% in the lead time.

1. はじめに

我々は，センサで車両周囲の環境を検知し，その結果を用いてステアリング操作を制御する超音波センサシステムの新製品開発を担当している．このシステムは，ユーザの実使用環境の影響を受け易いため，仕様開発と量産開発の２つのフェーズに分けて開発している．仕様開発フェーズでは，研究開発部門が実車両上にシステムのプロトタイプを構築し，要素技術と要求仕様を開発している．量産開発フェーズでは，製品開発部門が要求仕様を基に製品設計をして製品版ソフトウェアを開発している．

しかしながら，近年，自動車メーカからの車載システム機能の高度化，複雑化，短納期かつ高品質要求に応えるためには，以下の問題により従来の２フェーズ開発方式を保つことが難しくなった．

・ 仕様開発：要素技術開発に時間を要し，製品設計が求める要求仕様書の作成時間が確保できない．
・ 量産開発：仕様開発フェーズが作成する要求仕様書を待っていては量産開発期間が確保できない．

そこで，本論文では，以下に示す工夫により，従来の２フェーズ開発方式の前後二つのフェーズの一部を重ね合わせたコンカレント開発方式を導入し，製品開発期間を短縮することを提案する．

・ 製品開発部門にて，仕様開発フェーズのプロトタイプのソースコードから早期に製品設計を開始し，要求仕様を抽出して研究開発部門と知識を共有する仕組みを導入する．

このコンカレント開発方式を実際のプロジェクトに適用した結果，従来の２フェーズ開発方式で見積もった開発期間６ヶ月に対して，2.5 ヶ月と期間短縮することができた．また，改善後のプロセス工数を分析することで，さらなる納期短縮が望めることもわかった．

本論文では，２章でこれまでの製品開発プロセスを分析し，3，４章でコンカレント開発方式の考え方と具体的な施策を説明する．５章で実際のプロジェクトへの導入，6，７章で実施結果と考察を示す．最後に 8，９章でまとめと今後への改善点を述べる．

株式会社デンソー　走行安全技術４部
Driving Assist & Safety Eng. Div.4　DENSO CORPORATION
〒448-8661 愛知県刈谷市昭和町 1-1　Tel:0566-55-5810　e-mail:kengo_hayashi@denso.co.jp
1-1, Showa-cho,Kariya-shi,Aichi,Japan

2. これまでの製品開発プロセスの分析

我々の職場では，製品開発プロセスとして，仕様開発と量産開発に分けた，2 フェーズでの開発方式を採用している(図 1)．前章で述べた通り，顧客ならびに市場からは，開発期間を要する要求と，納期短縮を求める矛盾した要求が強くなっており，これまでの 2 フェーズ開発方式を保つことが難しくなった．

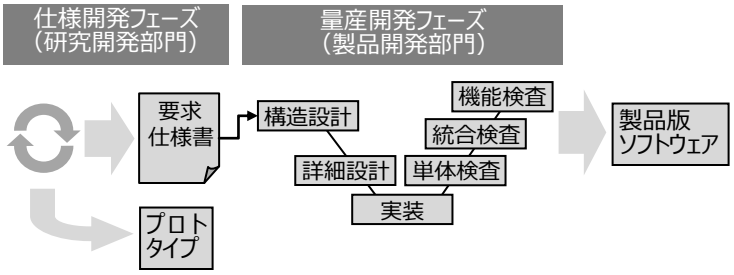

図 1 これまでの製品開発プロセス

その原因は，フェーズ間を要求仕様書の受け渡しによって移行するところにある．仕様開発と量産開発では，それぞれ必要とされる技術が異なる．組織と責務を分離し，定義された文書でフェーズ間を移行することは，品質を確保して後戻りを防ぐための仕組みとしては機能している．ところが，開発期間の短縮という観点では，フェーズ間を移行するために要求仕様書を整備する期間がボトルネックとなって，量産開発フェーズの開始が遅延し，開発期間そのものの遅延につながってしまう．

特に，新規機能開発においては，仕様開発の量が明確ではなく，機能の難易度も均一ではない．品質や後戻りを考慮したとしても，仕様開発のすべての完了を待つことは，期間リスクが大きい．一方，安定した仕様から部分的に要求仕様を文書化していく方法もある．しかし，プロトタイプ開発をして動作検証しながら開発を進めている研究開発部門の担当者にとって，開発と文書化の並行作業は工数と集中力の分散を招き，結果として要求仕様開発の全体完了を遅らせる要因となり得る．

以上より，製品開発の期間を短縮するためには，これまでの製品開発プロセスにおいて，ボトルネックとなる要求仕様書によるフェーズ移行に着目することが必要となる．

3. コンカレント開発方式の導入

前章で分析した通り，フェーズ間を要求仕様書の受け渡しによって移行することが，製品開発の短納期化の妨げとなっている．一方，要求仕様書に記述すべき内容は，研究開発部門の担当者の知識として，検討・検証に用いた資料として，或いは開発したプロトタイプのソースコード上の実装結果として分散して存在している．文書として整えられていなくても，これらのリソースが活用できれば，仕様開発フェーズの完了を待たずに量産開発フェーズを開始することが可能である．

そこで，製品開発部門において，研究開発部門における仕様開発フェーズのプロトタイプのソースコードをベースに，製品設計を早期に着手することで，製品開発期間を短縮する方法を検討することとした．このプロトタイプソフトは，仕様開発フェーズでは使い捨てとして車両メーカと共同で開発していることから，機能の妥当性確認や性能検証は果たしているものの，設計仕様が整備されておらず，ソフトの論理検証は不十分であることがわかっている．仕様を得ていない状態での量産開発部門の担当者では，設計・検証のために必要な，振る舞いや構造への知識が不足してしまう(図 2)．製品開発期間に延長影響を与えない形で，製品の要求仕様と設計仕様をプロトタイプのソースコードから抽出する仕組みを導入するこ

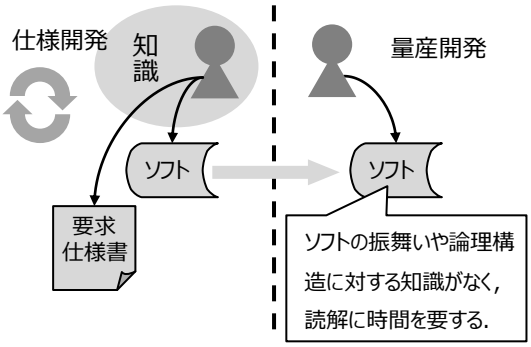

図 2 分担開発による知識の局在化

とが必要不可欠である．

これらを考慮して，仕様開発におけるプロトタイプを利用したコンカレント開発方式のプロセスを設計した(図 3)．プロトタイプを流用したときの知識不足を補うことと，開発開始時期を早めることを狙って，量産開発フェーズの担当者で製品の要求仕様と設計仕様をプロトタイプのソースコードから抽出する工程を追加した．また，静的解析ツールを用いてソースコードを修正する工程を始めに配置した．ソースコードに潜在する欠陥は，設計仕様を解釈するときに誤りを誘発する要因となり得るため，予めこれを除去し，後の修正工数を削減するのが狙いである．さらに，設計仕様を抽出した時点で，プロトタイプにおける設計とソースコードの流用可否を判断し，流用可能と判断されれば，量産開発フェーズでそのまま再利用，或いは変更を加えた上で再利用し，再開発のムダを省くようにした．仕様抽出の具体的な施策については、次章にて示す．

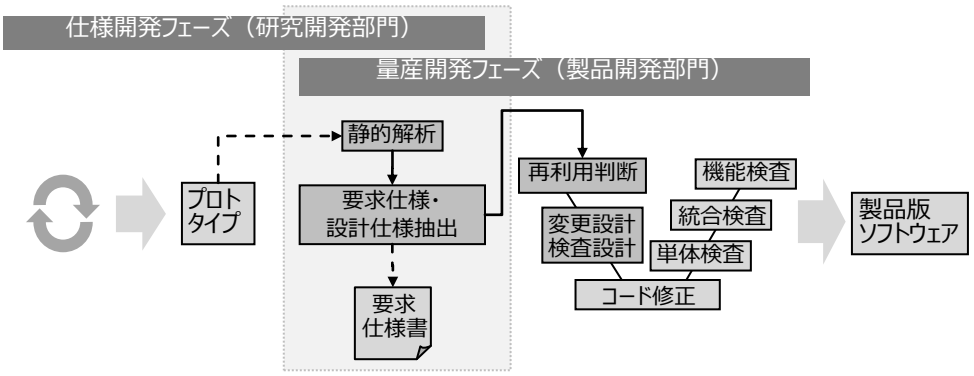

図 3 プロトタイプを利用したコンカレント開発方式

仕様開発フェーズのプロトタイプ開発までと，量産開発フェーズのコーディング以降の工程は従来通り維持している．但し，量産開発フェーズの各検査設計工程は，人的資源・検査環境数の許す限り同時並行での着手を許した．検査設計は，より上流で見つかる欠陥による後戻りを防ぐため、順次実行するのが通常である．しかし，再利用元となるプロトタイプは実動作環境で動作確認されていることから，重大な後戻りを引き起こす欠陥は少ないと見積もれる．順次実行が元で期間が積み上げられて，開発期間が延びる方がリスクが高いと考え，同時並行で検査設計する方針とした．

以上の考えを基に，仕様開発におけるプロトタイプを利用したコンカレント開発方式による製品開発プロセスを設計した．

4. プロトタイプからのソフト要求仕様・設計仕様の抽出

ソフトウェアの検証のために必要な，振る舞いやソフトウェア構造への知識を補うために，製品の要求仕様と設計仕様をプロトタイプのソースコードから抽出する仕組みを導入する．

部品を再利用するための手法はリバースエンジニアリングの手法として幾つも知られている[1][2]．その中に，知識の乏しいプロジェクトに対して変更要求仕様

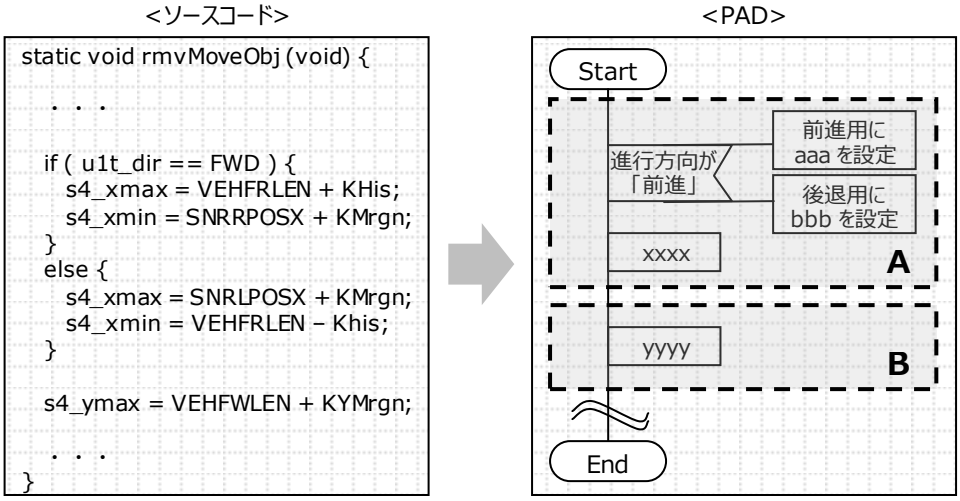

図 4 ソースコードから PAD への変換例

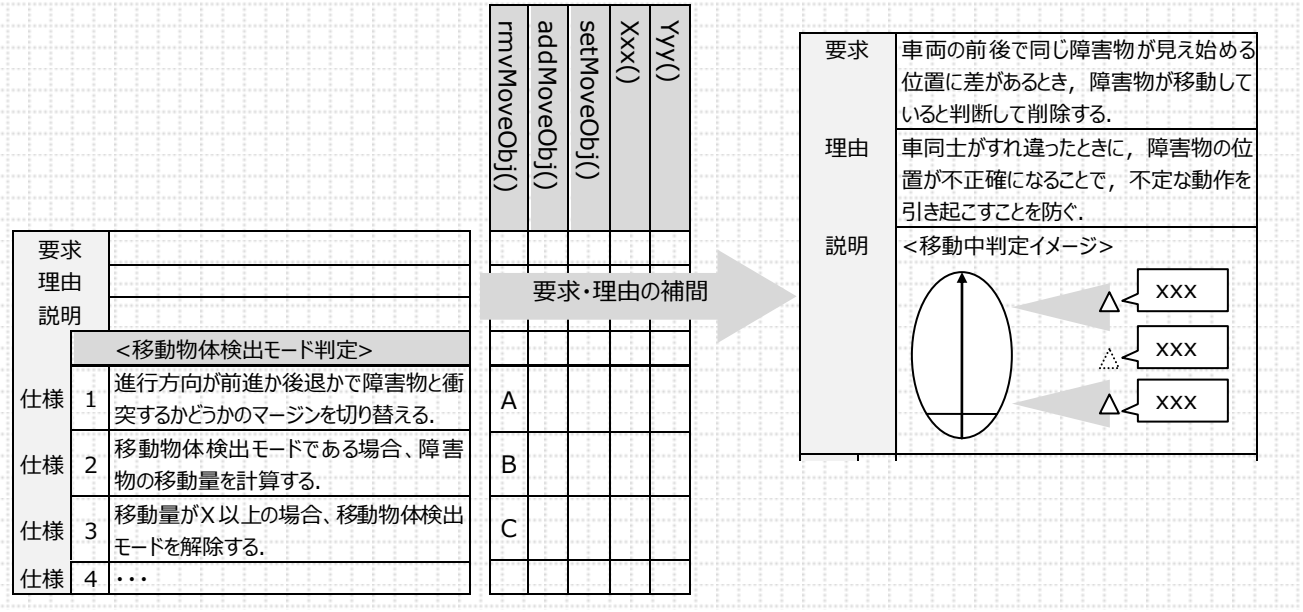

図 5 USDM による要求仕様記述と要求・理由欄補間例

書を作成する方法として，ソースコードを基に，DFD(Data Flow Diagram)や AFD(Architecture Flow Diagram)などのモデルを経由して，USDM(Universal Specification Describing Manner)を使って記述する方法が提案されている[3][4]．本論文では，利用するモデルとして，設計仕様書を PAD(Problem Analysis Diagram)で記述し，要求仕様書を USDM で記述する方法を採用した．

PAD を採択したのは，ソースコードの構造と記述の構造が同一で対応を取り易く，ソースコードからの生成が容易なためである．また，構造が上から下に一方向にだけ流れることで，一連の処理のまとまりを分割して括りやすく，グループ化させたときに識別し易い（図 4 参照）．

USDM を採択したのは，PAD と同様に構造的に仕様を扱うことができ，トレーサビリティ・マトリクスを備えており,PAD で識別した処理のグループとの対応を記録できるためである(図 5 参照).対応を記録することで，要求仕様と設計仕様，ソースコードの対応を追うことができ，設計の正当性の確認や，仕様抽出における抜け・漏れがないことを確認することができる．要求仕様を導出する上で，振る舞いを理解するために，DFD や状態遷移図を利用した場合，これも PAD と合わせて設計仕様として残しておく．

USDM で記述された仕様は,開発機能に対する知識が少ない量産開発フェーズの担当者によって記述されるため,その解釈に誤りを含んだり,解釈の過程でロジックの意図そのものが理解できなかったりすることがある．特に，仕様によって実現したかった要求そのものや，要求が発生した理由，アルゴリズムを採用した理由は仕様開発フェーズの担当者の頭の中に暗黙知化されている．これは，仕様開発の過程で要求分析が進み，要求仕様が開発されているためである．そこで，仕様開発フェーズの担当者によって，仕様解釈に誤りがないかを確認すると共に，要求仕様書の要求と理由を記述するようにした(図 5 参照)．その上で，再度量産開発フェーズ担当にて仕様が要求と理由に合致しているかを確認した．

以上により，製品の要求仕様と設計仕様を，プロトタイプのソースコードから量産開発フェーズの担当者が抽出して文書化し，仕様開発における暗黙知を形式知化して共有した．

5. 超音波センサシステム開発プロジェクトへの導入

これら具体的施策を，超音波センサ応用システム開発プロジェクトに導入した．本システムは，超音波セン

サを用いた車両制御を伴う新システム開発であり，複数の機能を独立したコンポーネントで実現する．これらの内，規模の異なる 2 つのコンポーネントを選択し，コンカレント開発方式による製品開発プロセスの試行対象とした．仕様開発において開発されたプロトタイプのソースコード行数から，標準的な開発期間を表 1 のように SLIM[5]にて見積もった．選択したコンポーネントに対して，コンカレント開発方式に従い，2 つのフェーズを 0.5～1 ヶ月の期間を重ねて導入着手した．

表 1 SLIM による開発期間の見積もり

	開発コンポーネント	
	A	B
規模	14KLOC	5.4KLOC
生産性指標(社会平均を入力)	13.0	13.0
組織構築率(Max 値を入力)	5	5
最短開発期間(月)	6.05	4.02

フェーズを重ね合わせる要求仕様・設計仕様抽出プロセスは，新たな工程となる．本プロセスにおける冗長なコストと，従来工程へ生じる副作用を確認するため，各工程に費やした工数を記録し，結果として分析する．

6. コンカレント開発方式の適用結果

(1)開発期間の短縮

開発期間の短縮効果を測る指標として，図 6 に開発コンポーネントの開発期間の見積もりと実績の比較結果を示す．このように，開発期間は 46%～59.7%短縮され，施策の効果が確認できた．

本結果は，量産開発フェーズに相当する開発期間単独で比較したが，施策では量産開発フェーズの開始を 1 ヶ月前倒しして開始している．量産開発期間を仕様書発行期間からのみの期間とすると，それぞれ 1 ヶ月ずつ早まることから，開発期間は 62.8%～84.6%短縮されており，量産開発フェーズは 4 割以下の開発期間で終えている．

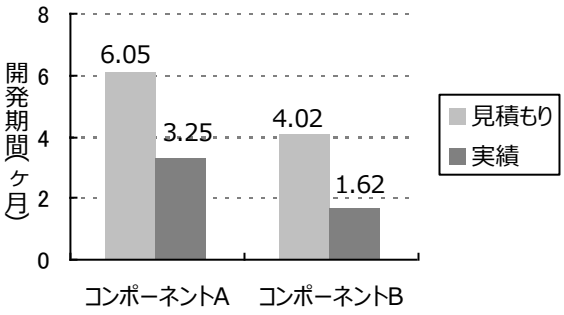

図 6 コンポーネント開発期間の比較結果

これまでの製品開発プロセスに対して，コンカレント開発方式では，4 ヶ月弱納期短縮する効果が得られたことがわかった．

(2)量産開発フェーズの作業工数の割合と欠陥抽出の傾向

今回の試行では，知識共有の効果を高めることを目的に，量産開発フェーズ担当者にてソースコードから PAD への変換を手作業で行った．ソースコードを参照する機会を増やして理解度を高めることが狙いだったが，量産開発フェーズに占める設計仕様化の工数を分析したところ，図 7 で示される通り，19.7%～27.7%と高い比率を占めていた．

設計仕様化のプロセスにおいて，高い工数比率以外の特徴が現れていないかを確認するため，量産開発フェーズで抽出する欠陥を，実際に製品の不具合として発症する機能性欠陥と，誤記やコードスタイル不統一など，直接は不具合とならないが今後欠陥を引き起こす誘発要因となる発展性欠陥に分類し，それぞれプロトタイプに埋め込まれた欠陥か，量産開発で成果物を生成したときに埋め込まれた欠陥かを分類して集計した(図 8)．その結果，量産開発フェーズで埋め込まれる発展性欠陥の割合が 16.6%～24.0%と高いことがわかった．このほとんどは，設計仕様化で手作業を行ったことによる，記述ミスである．

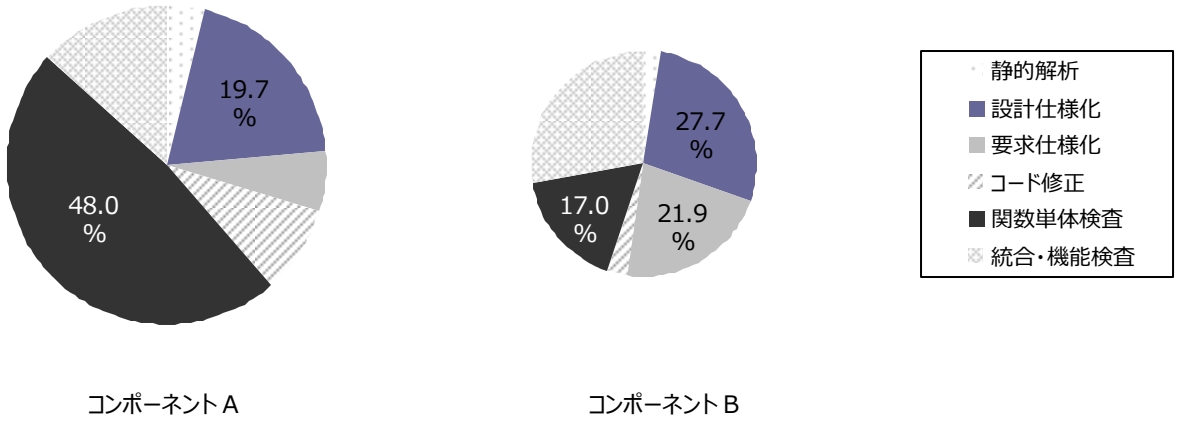

図 7 量産開発フェーズの作業工数の割合

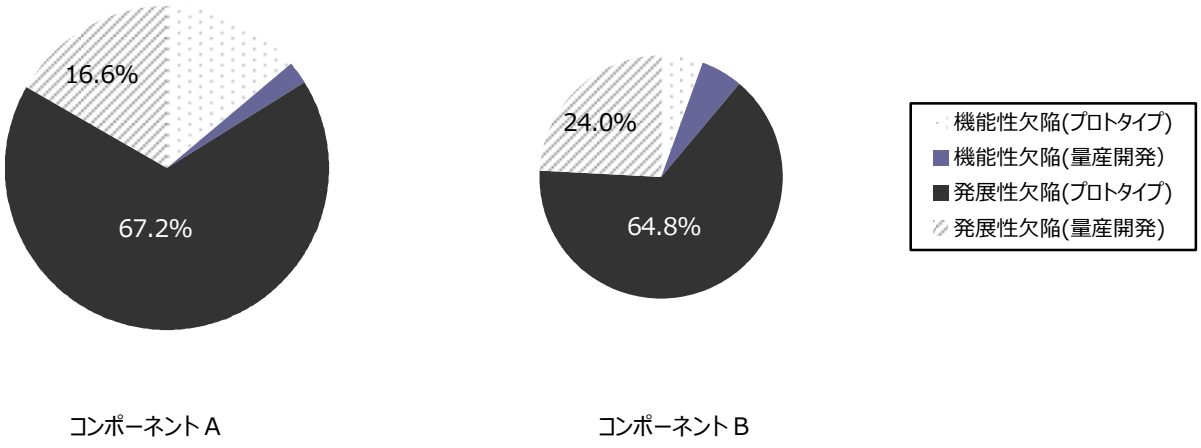

図 8 量産開発フェーズで抽出した欠陥数の割合

また，開発における関数単体検査が占める比率も 17.0%～48.0%と高いことがわかった(図 7)．それぞれ次章にて考察を示す．

7. 考察

(1)期間短縮結果に対する考察

今回の導入では開発期間を 4 ヶ月弱短縮することに成功した．その要因を，従来の 2 フェーズそれぞれに対して考察する．

仕様開発フェーズでは，要求仕様書の記述作業が部分的に省けるようになっただけで，プロセスの変化は小さい．主な影響は，要求仕様に対する疑問点や不明点の発生が，プロトタイプのソースコードに対して発生するようになったことである．その結果，問い合わせの内容が細かく数が多くなったが，要求仕様書の生成工数が一部省けて余力が生じていたことで，開発遅延にはつながらなかったと推察される．

量産開発フェーズでは，下流工程のプロセスは従来通りである．プロトタイプのソースコードを基に仕様抽出するプロセス追加が大きな変化点であるが，これらの作業は製品を派生開発するときに採用しているプロセ

スを応用したものであり，量産開発フェーズの担当の経験の範囲内で適応できたため，開発遅延につながらなかったと推察できる．

今回の導入では，コンポーネントの部品と設計の多くを再利用できたことも，期間短縮できた大きな要因として挙げられる．仕様開発フェーズのプロトタイプ開発はインクリメントを重ねる中で，アーキテクチャを安定させていた．先に示した通り，コンポーネントAの変更影響は生じたものの，アーキテクチャや大枠の設計はほとんど再利用できており，変化量を小さく抑えられたことで，開発工数・期間も抑えられている．コンカレント開発により期間短縮を果たす上で，プロトタイプの品質は特に注意すべき観点である．

(2)設計仕様化工数の妥当性

前章の分析結果の通り，設計仕様化工程が占める工数はその他工程に比べて高い．特にコンポーネントAでは，開発総工数そのものが大きいため，相当するコストも高かった．コンポーネントAでは，プロトタイプが安定するのが遅く，変更による手戻りが繰り返されたことで工数を増やす要因となった．しかし，設計仕様化の工数が元々小さければ，作業開始時期を遅らせることができ，変更による手戻りは軽減できる．もしくは，変更する可能性の高い部品は設計仕様化の優先度を落とすなどの対策も考えられる．

設計仕様化工程で流入する発展性欠陥の割合も高かった．プロトタイプは実動作環境で評価しているため，機能性欠陥の割合が小さく，発展性欠陥の割合が大きいことは想定通りである．ところが，量産開発フェーズで埋め込まれた発展性欠陥のほとんどは，設計仕様化を手作業で行ったことによる，記述ミスである．これらの発展性欠陥は，製品の不具合としては検出されないものの，レビュー後の手戻りのムダや，検査設計時に誤りを誘発させるリスクにつながる．

設計仕様化を手作業で行うのは，知識共有の促進を狙ってのことであったが，実際には，その後の要求仕様化や，機能検査設計や統合検査設計でもソフトウェア構造を理解する機会は多い．設計仕様化は何らかのツールを用いて自動化し，その後の作業で知識共有していく方法も検討すべきと考えられた．

(3)関数単体検査工数の費用対効果

関数単体検査は,製品の安全を保証するために,テストカバレッジを満たすべく時間を掛けている.しかし,プロトタイプは実動作環境での評価実績があることから,関数に存在する多くのパスは実行済みである割合が高い．改めてすべてを関数単体検査するのはムダが大きく，費用対効果も低い．関数単体検査の目的と効果を鑑みて方法を見直すことで，開発コストと必要期間はさらに短縮できると見込まれる．

8. まとめ

車載システムの開発期間を短縮するために，これまでの開発プロセスの現状を分析した．2つの施策を実施して，使い捨て型プロトタイピングによる2フェーズ開発から，2つのフェーズの一部を重ね合わせたコンカレント開発方式を試行した．その結果，製品開発期間を４ヶ月短縮することに成功した．また，仕様開発におけるプロトタイプを流用してコンカレント開発を導入したとき，プロトタイプからの再利用率が高ければ，開発期間短縮の効果が高くなると推察できた．

9. 今後の進め方

本方式による開発期間の短縮効果を高めるために，今後以下2つの施策を検討する．

(1) 設計仕様化の自動化

設計仕様化を手作業で行うのは多くのムダを生じさせる．その後の要求仕様化や検査工程を鑑みて，ソースコードから設計仕様を抽出する最適な自動化手法，或いはツールを検討する．

(2) 関数単体検査の効率化

関数単体検査にも多くの工数が費やされている．品質保証戦略を見直し，関数単体検査に相当するが工数のより小さい手法を検討する．

また，本論文は初めての導入試行であり，新機能開発を前提としている．今後は以下 2 つの施策を検討する．

(1) より早期での重ね合わせの開始

仕様開発のより早い時点から，仕様開発の障害とならないよう，製品をリリースするための必要作業の成果を残すことを焦点にプロセス設計し，さらなる開発納期の短縮を狙う．

(2) ソフト派生開発への追従

仕様開発は次期製品開発に向けて，或いは同一製品の試作フェーズの次納入に向けて継続される．プロトタイプも派生開発されることになるため，派生開発でもフェーズを重ね合わせて追従できるようプロセスを設計する．

参考文献

[1] Biggerstaff, T. J., ”Design Recovery for Maintenance and Reuse”, IEEE Software, Vol.22, No.7, pp.36-49, 1989

[2] Chikofsky, E.J.and Cross,J.H., Ⅱ, ”Reverse Engineering and Design Recovery”, A Taxonomy, IEEE Software, Vol.7, No.1, pp.13-17, 1990

[3] 津田剛宏, ”設計手法を活用した変更要求仕様書の作成手法 -完全無知見プロジェクトへの XDDP の適用-“, SQiP, 2010

[4] 中井栄次, ”無知見プロジェクトに対する XDDP の適用”, SQiP, 2009

[5] L. H. Putnam, “A General Empirical Solution to the Macro Sizing and Estimating Probrem,” IEEE Trans. on Software Eng., Vol.4, Mo.4, pp.345-361, 1971.