

自動車組込みソフトウェアへのサービス指向アーキテクチャの提案

田邊 隼希[†] 生駒 光平[†] 近藤 貴大[†] 中道 上[†] 青山 幹雄[†]

南山大学 数理情報学部 情報通信学科[†]

1. はじめに

複数ベンダで開発した自動車組込みソフトウェアのネットワークを介した協調制御が必要である。本稿はサービス指向アーキテクチャ(SOA:Service-Oriented Architecture)による実現方法を提案する。

2. SOA 適用の問題点

自動車などの組込みソフトウェアでは、イベント処理が重要であるが、従来の SOA はイベントを扱えない。サービスはユーザが直接結果を受けると、空間を介して間接的に効果を得るものも扱う必要がある。

3. アプローチ

3.1. プロパティ中心サービス提供モデル

SOA を適用する上でプロパティを中心にモデル化を行う[1]。プロパティを次の三つに分類する。

- (1) サービスプロパティ：サービスの特性や状態。
- (2) 環境プロパティ：ユーザ周辺の環境の状態。
- (3) ユーザプロパティ：ユーザの特性や要求。

3.2. モデルに基づく SOA 設計方法

サービス提供を様々な観点から分類し、それを実現するアーキテクチャパターンを決定する。

サービスが制御するプロパティの性質とユーザとの相互作用に着目し、次の 2 種類に分ける。

- (1) 対話型サービス提供：ユーザの要求に対しプロパティを動的に変更する。
- (2) 自律型サービス提供：プロパティ値を一定に保つためシステムが自律的にサービスを提供する。

3.2.1. アーキテクチャパターンの決定と統合制御

対話型サービス提供では、ユーザはスイッチ操作などでサービスを直接指定、起動し、結果を待つ。従ってメッセージベースの要求/応答を用いる。

自律型サービス提供では、環境の情報をイベントとして検出し、関連するサービスを起動する。イベントの発生とそれに応じた処理を独立に行うため、イベントベースのパブリッシュ/サブスクライブを用いる。

この二つのアーキテクチャパターンは振舞いが異なるが、システム全体では統合制御する必要がある。

4. ブローカーアーキテクチャ

アーキテクチャパターンを統合制御するブローカーアーキテクチャを提案する。

ブローカをイベント処理とサービス起動に分離する(図 1)。以下に詳細を示す。

A Service-Oriented Architecture for Automotive Embedded Software Systems

[†]Hayaki Tanabe, Kohei Ikoma, Takahiro Kondo, Noboru Nakamichi, Mikio Aoyama, Nanzan University

(1) イベントマネージャ：センサからのイベントを受信し、登録されたサブスクリプションに基づきフィルタリングを行い、サービスコーディネータにイベントを配信する。

(2) サービスコーディネータ：イベントマネージャとの連携やユーザ、サービスからの要求に基づきサービスを起動する。

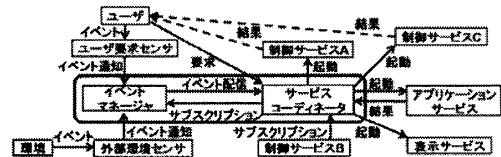


図 1 ブローカーアーキテクチャ

5. 提案アーキテクチャの適用と評価

提案アーキテクチャを実際の ACC(Adaptive Cruise Control System)[3]に適用し、有効性を評価した。比較のために、従来 ACC システムと従来の SOA による ACC の実現例を同様に分析した。

5.1. 提案アーキテクチャの適用例

5.1.1. 機能分析(ユースケース分析/シナリオ分析)

ACC の提供機能とアクタの関係を示す(図 2)。これに基づき、ACC の開始から追従走行制御までのシナリオを作成した。

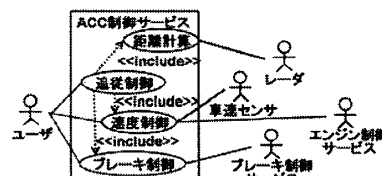


図 2 ACC のユースケース図

5.1.2. 相互作用分析

(1) 実行順序に着目した相互作用分析

ACC の相互作用を分析するために、シナリオに基づきシーケンス図を作成した(図 3)。

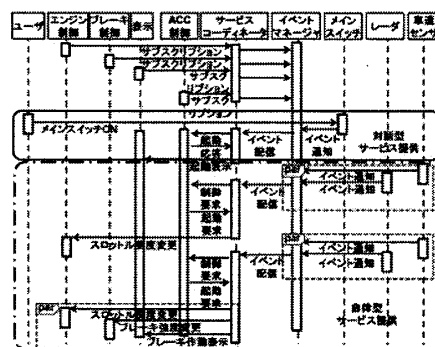


図 3 ACC のシーケンス図

シーケンス図から次の二つのサービス提供パターンを発見した。

1)対話型サービス提供パターン：ユーザ要求により ACC 制御サービスを起動。

2)自律型サービス提供パターン：環境プロパティの変化により ACC 制御サービスが起動され、プロパティの値を保持する制御。

(2) 要素間の関係に着目した相互作用分析

図 3 のシーケンス図より、コミュニケーション図を作成し、要素間の相互作用を分析した(図 4)。

イベントはイベントマネージャに通知され、サービスコーディネータに配信される。サービスコーディネータはイベントやサービスの要求に基づきサービスを起動する。サービスはサービスコーディネータにより起動され、サービス間で直接メッセージ交換はしない。提案アーキテクチャによりイベント処理とサービス起動を分離できた。

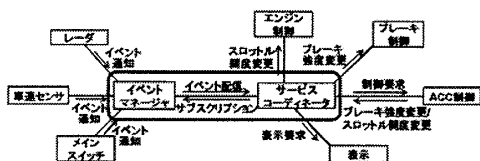


図 4 ACC のコミュニケーション図

5.2. 従来 ACC システムの分析

比較対象として従来 ACC システムを同様に分析した。

5.2.1. 機能分析(ユースケース分析/シナリオ分析)

従来 ACC システムの機能(図 5)に基づき、ACC 開始から追従走行制御までのシナリオを作成した。

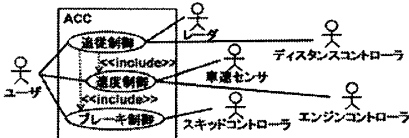


図 5 従来 ACC システムのユースケース図

5.2.2. 相互作用分析

シナリオに基づき、コミュニケーション図を作成し要素間の相互作用を分析した(図 6)。

コントローラ間、センサ - コントローラ間で直接イベントを交換しており、結合が強い。機能やセンサの追加、変更には関連するコントローラの変更が必要となる。センサのイベントはあらかじめ指定されたコントローラにしか配信できない。

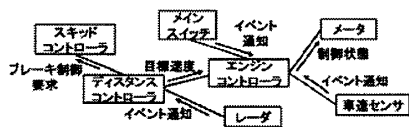


図 6 従来 ACC システムのコミュニケーション図

5.3. 従来の SOA による ACC の分析

従来の SOA ではイベントを扱えないため、ACC を実現した場合、車速センサ、レーダのイベントにより ACC 制御サービスは起動できない。また、エンジン制御サービスやブレーキ制御サービスの実行結果は間接的にユーザに作用し、従来の直接的な(要求/応答型)サービス提供形態とは異なる。

5.4. 提案 SOA の比較、評価

従来システム、従来 SOA と提案 SOA を SOA の特性、イベント処理、プロパティ制御の観点から比較を行い、評価した(表 1)。

表 1 ACC によるアーキテクチャの評価

観点	評価項目	従来システム	従来 SOA	提案 SOA
SOA の特性	インタフェース形式	ベンダごとに異なる	標準化されている	標準化されている
	プラットフォーム依存性	依存	非依存	非依存
イベント処理	イベント処理の可能性	可能	不可能	可能
	イベントの配信制御	固定的	イベントの処理が不可能	追加・変更が可能
プロパティ制御	サービス提供形態	サービスは存在しない	直接的サービス提供	直接的・間接的サービス提供

(1) インタフェース形式

提案 SOA ではインタフェースの標準化により、サービスの再利用性が向上し、車内のサービスと車外のサービスの連携が容易となる。

(2) プラットフォーム依存性

提案 SOA はプラットフォームに非依存で、プラットフォームが異なるベンダ間で開発されたサービスの連携が容易となる。

(3) イベント処理の可能性

提案 SOA はイベントマネージャとサービスコーディネータの連携により、イベント処理が可能となる。

(4) イベントの配信制御

提案 SOA はパブリッシュ/サブスクライブを用いてイベント配信先の追加と変更が可能となり、イベントを複数サービスに配信できる。

(5) サービス提供形態

ACC ではサービスの実行結果が間接的にユーザに作用するが、提案 SOA ではプロパティに着目し制御を行うため、間接的なサービス提供が可能になる。

以上の議論より、提案 SOA は自動車組込みソフトウェアの実現に有効なアーキテクチャといえる。

6.まとめ

従来 SOA の直接サービス提供に加え、イベント起動と間接サービス提供を含むプロパティ中心サービス提供モデルを示し、実現するためにブローカーアーキテクチャを提案し、有効性を評価した[2]。今後リアルタイム性について研究を進める。

参考文献

[1] 青山 幹雄, 藤山 麻衣, 組込みネットワークシステムのユニバーサルサービス指向アーキテクチャ, SES 2008 論文集, Sep. 2008, pp. 147-154.
 [2] 生駒 光平, 近藤 貴大, 田邊 隼希, 自動車ソフトウェアにおけるサービス指向アーキテクチャの提案, 南山大学 2008 年度卒業論文, 2009.
 [3] トヨタ自動車, CROWN MAJESTA 新型車解説書, 2004.