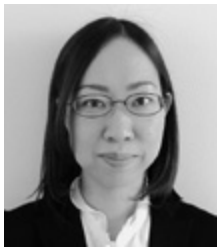
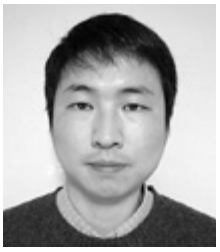
自動運転車を取り巻く System of Systemsの安全性要求の妥当性確認と検証






木下 聡子※　西村 秀和※　ユン ソンギル※　北村 憲康※

より安全な交通環境の実現を目指し，2020年を目処に自動運転車を社会へ導入するための取り組みが行われている．しかしながら，自動運転車自体で安全性を実現することは困難であり，周辺システムと協働することにより全体の安全性を確保することが重要となる．本論文では，自動運転車とその周辺システムを含む交通システム全体をSystem of Systems（SoS）と見なし，そのシステムモデルに基づき導出された安全性要求をもとにGSN（Goal Structuring Notation）によりアシュアランスケース記述を行うことでその妥当性確認を行うとともに，モデル検査を用いて検証を行う．モデル検査では，SoSの構成要素である自動運転システムとドライバ間の相互作用に関する安全性要求を検証するためCSP（Communicating Sequential Processes）モデルを用いる．

Validation and Verification of Safety Requirements for the System of Systems involving Automated Vehicles

Satoko Kinoshita, Hidekazu Nishimura, Sunkil Yun, and Noriyasu Kitamura
Graduate School of System Design and Management, Keio University

There are efforts to introduce automated vehicles for the realization of safer traffic environments. However, this cannot be achieved just by introducing automated vehicles into the environments. Accordingly, it is necessary to ensure the safety of a traffic environment considering a traffic system including automated vehicles and the surrounding systems. This paper treats those systems as a system of systems (SoS) and introduces the system models. First, safety requirements derived from the system models are validated with descriptions of assurance cases. Second, by using model checking for a CSP (Communicating Sequential Processes) model, we verified that the system model satisfies the safety requirements relating to interactions between the automated driving system and the driver, which are constituent systems of the SoS.

1　はじめに

2020年に自動運転車を社会に導入することを目指し，自動車会社やIT関連企業が自動運転車の開発を進めている．内閣府が進める戦略的イノベーション創造プログラムでは，交通事故の死者数の低減及び交通渋滞を緩和することを目指し，自動走行システムに関する技術開発や国際連携に向けた準備が進んでいる［内閣府2016］．自動

※慶應義塾大学　システムデザイン・マネジメント研究科

運転車を含む交通システムは，多様な利害関係者を巻き込むこととなるため，自動運転車の社会受容性を検討する必要があると考えられる．

［BCG 2015］によると，自動運転システムへの最大の懸念は安全性にある．社会受容性にとって安全性は極めて重要な要素である．しかし，自動運転システムを搭載する自動運転車の安全性が独立して成り立つことはない．それは，情報システムや交通インフラなどと連携して動作することによって確保できるものである．すなわち，自動運転システムの安全性要求を明確にするためには，交通システムを構成するその他の独立したシステムと協働して動作をすることを前提に検討する必要がある．自動運転車を取り巻く情報システム，交通インフラシステム，歩行者，自動車や自転車などのモビリティ，道路や障害物などの物理環境，天候や高度などの自然環境はそれぞれ独立して動作している．更に，地理的に離れた場所に存在するため，このようなシステム全体はSystem of Systems（以下，SoSと略す）として定義できる［Jamshidi 2011］．［DeLaurentis 2005］は既に交通システム全体をSoSとみなすことを提案しており，自動運転車の安全性を検討する際にもSoSの問題として捉えることで，自動運転システムに関係するシステムやそれらの相互関係を明確に検討することが可能となる．

COMPASS（Comprehensive Modelling for Advanced Systems of Systems）［COMPASS］では，SoSに対してモデル検査を行い，構成システムの相互作用の結果として現れるSoSの創発的な振る舞いを検証する研究が行われた．そこでは，VDM（Vienna Development Method）とCommunicating Sequential Processes（CSP）に基づく独自の形式仕様記述言語（CML，COMPASS Modeling Language）を提案している．CSPは並行システムを形式的に記述し，検証するための理論である［Shneider1999］．

本論文では，自動運転車とそれを取り巻く交通システム全体をSoSとみなし，SoSアーキテクチャを構築する過程での安全性要求の妥当性確認と検証を行う．自動運転システムの自動化レベルとしては，SAE International（Social of Automotive Engineers International）の定義［SAE2014］でレベル３を仮定する．このレベルでは自動運転システムがドライバに運転権限の移譲を要求する際には，ドライバがそれに応答する必要がある．このため，本論文では自動運転車のドライバと自動運転システムとのコミュニケーションが安全性に大きく関与するものと考え，双方の状態変化及び相互作用を対象とする．まず，SysML（Systems Modeling Language）［Friedenthal2014］を用いたシステムモデル記述［西村2016］，［ユン2016］に基づき導出された安全性要求をもとにGSN（Goal Structuring Notation）によりアシュアランスケース記述を行うことでその妥当性確認を行う．そして，SoSアーキテクチャで定義されるドライバと自動運転システムの状態変化と相互作用をCSPモデルで表現し，妥当性確認をした安全性要求が満たされることをモデル検査により検証する．

2　SoSアーキテクチャの検証

2.1　自動運転車を取り巻くSoSアーキテクチャ

自動運転車が導入される交通環境には，渋滞情報や地図情報などを提供する情報システム（ICT System，Information Communication Technology System），信号機や道路標識などの交通インフラシステム（TIS，Transport Infrastructure System），ほかの車両（Surrounding Mobility），歩行者（Pedestrian），障害物などの物理環境（Physical Environment），天候などの自然環境（Natural Environment）など様々な独立したシステムが存在する．自動運転システムはこれらの独立したシステムと相互作用することで動作し，安全性を実現する必要がある．文献［西村2016］では自動運転システムとその周辺システム全体を自動運転車を取り巻くSoSとして見なし，全体として安全な交通システムを保障するためのアーキテクチャ定義の記述にシステムモデルを用いた．そこでは安全性を検討するために現状の交通事故をもとにユースケース分析を行い，SoSの構成システム間の関係性を構造と振る舞いのシステムモデルで示すと共に，安全性に関する機能要求，インターフェース要求を定義した．

2.2　安全性要求の妥当性確認と検証のアプローチ

SoSアーキテクチャを定義するため，導かれた安全性要求の妥当性確認と，安全性要求に基づくアーキテクチャの検証を行う．本節では，そのアプローチについて述べる．

自動運転車を取り巻くSoSの安全性を確保するというミッションに対する安全性要求の妥当性確認に際して，本論文では，アシュアランスケース［山本2014］を用いる．アシュアランスケースの記述により，システムモデルから得られる安全性要求をゴールとして設定し，議論をすることでそれらの妥当性を確認する．議論の論理展開を記述として残すことができるアシュアランスケースは，様々なステークホルダやシステム間の合意形成が必要なSoSの妥当性確認の方法として有効である．既に文献［Goodger2012］では，重要な情報基盤セキュリティの脆弱性を評価するため，評価再検討サイクルとアシュアランスケースの記述の併用を提案している．

本論文では，SoSアーキテクチャを構築する初期段階で，システムモデルをもとにアシュアランスケースの記述を行い，要求の妥当性確認を進める方法を提案している．また，システムモデルで明確にされる安全性要求に関係するSoSの構成システムを関連付けるため，［Saruwatari2014］で定義されているアクターというノードを導入し，SoSの構成システムをこのアクターとしてアシュアランスケースを記述する方法を提案する．SoSでは，各構成システムの相互作用の結果，安全性要求が達成されるか否かを検討することが重要である．構成システムをアクターとする記述方法を用いることにより，ゴールの達成に必要な構成システムを明確にできる．これにより，関連する各構成システムの運用者や利害関係者でゴールの達成に関する責任を議論することが可能となる．

次に，妥当性確認をした安全性要求に対し，SoSアーキテクチャがこれらの安全性要求を満たすことを検証する．SoS

の構成システムは独立に動作し，相互作用することによってSoS全体の機能が実現する．従って，SoSの構成システムの振る舞いと相互作用をCSPモデルで表現することによって，安全性要求を満たさない振る舞いのパターンを検出する．著者らは文献[Kinoshita2015]で，自動運転システムとドライバの相互作用を表すCSPモデルを提案した．本論文では，これにSoSアーキテクチャで追加された振る舞いを加え，かつ自動運転システムとドライバの状態が周辺システムから得られる情報によって変化することをCSPモデルに導入する．状態の変化が振る舞いと並行して起こることを表現することにより，より詳細な安全性要求を検証できる．

3 アシュアランスケースを用いた安全性要求の妥当性確認

3.1 SoSのアシュアランスケース記述

自動運転車を取り巻くSoSの相互接続の検討をもとに安全性に関する議論を行った結果を図1に示す．図1の最上位のゴールとして「自動運転システムがドライバと共に安全な運転を実現する」を設定する（ゴール：G_1）．このゴールを詳細に分析する戦略として，運転の基本動作である認知･判断･操作に基づいて論じるという戦略を設定した（戦略：S_1）．これは，自動車事故の多くはヒューマンエラーによるものであり，ヒューマンエラーが原因となる事故を減少させることで安全性を高めることができると判断したためである．

図1に示すアシュアランスケースで自動運転システムへの要求として，ドライバが通常の状態ではないことを検知することが求められ（ゴール：G_7），ドライバの状態を回復することが要求される（ゴール：G_8）．自動運転システムの働きかけによってもドライバの状態が回復しない場合は，自動運転システムが交通事故を起こす危険を回避する必要がある（ゴール：G_9）．これらの議論の結果より，自動運転車の導入により安全性を実現するためには，ドライバの状態を正確に把握する必要があり，必要に応じて交通事故を起こす危険を回避することが要求されることが明らかとなった．とくに，レベル3の自動運転システムの定義で最終的な責任を負うドライバを安全な状態に回復させるという機能はSoS全体の安全性を実現するために重要な要求である．

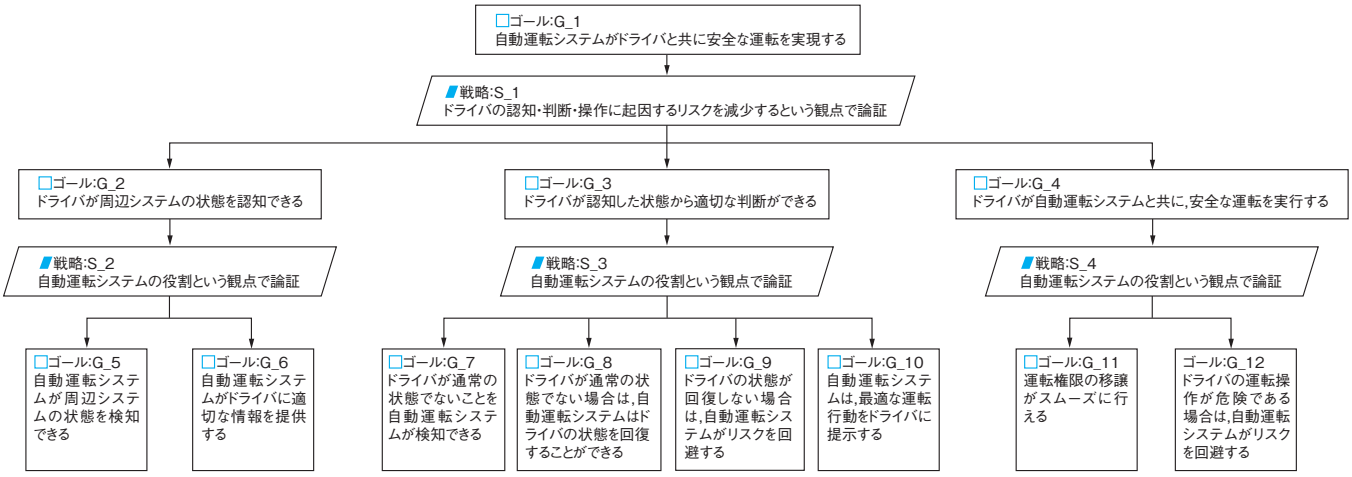


図1　自動運転車を取り巻くSoSの相互接続の検討をもとにしたアシュアランスケース

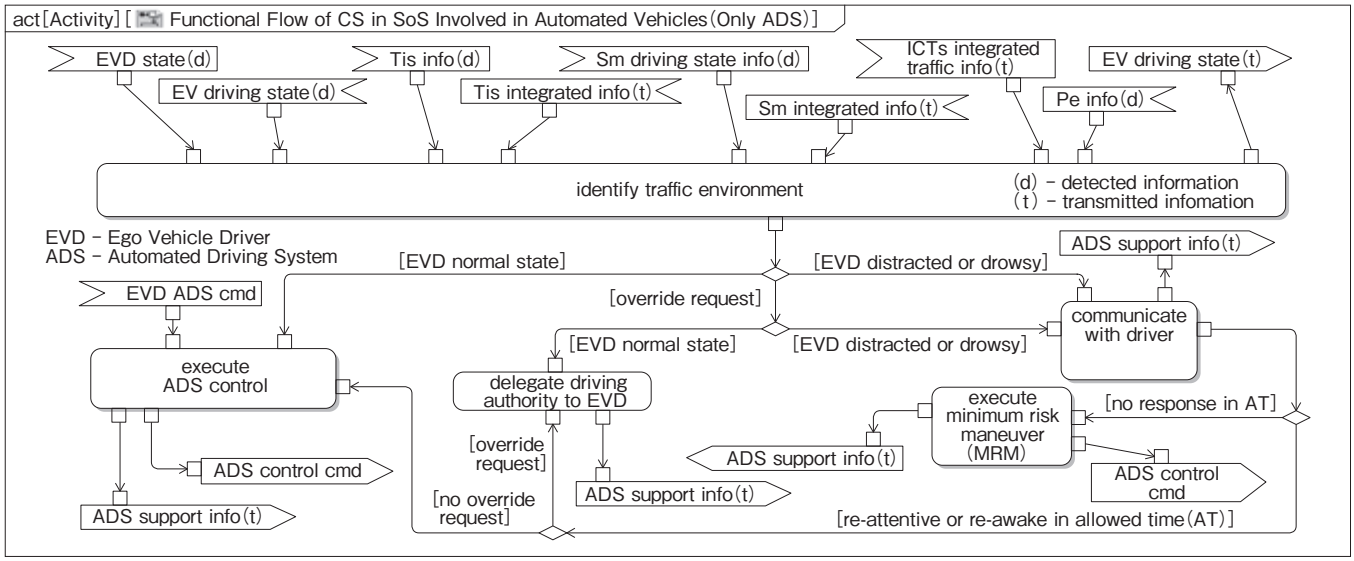


図2　自動運転システムの機能フローを示すアクティビティ図

3.2 アクターを用いた構成システム表現の導入

SoSの構成システム間の相互接続の検討ののち，ユースケース分析やインターフェースの検討の結果として，自動運転車を取り巻くSoSの機能の流れを表すアクティビティ図を記述した（図２）．このアクティビティ図より自動運転システム（Automated Driving System，ADS）に求められる安全性要求をアシュアランスケースにより記述する（図３）．このアシュアランスケースを用いて，図１に示す自動運転システムへの要求が図２より導かれる要求に引き継がれることを確認することによって，図２から導かれる安全性要求の妥当性を保証する．また，ゴールの達成に関連する構成システムをアクターのノードを用いて表現する．アクターは，図３のモジュールというフォルダの矩形で示されている．

まず，最上位のゴールとして「自動運転システムがドライバと共に安全な運転を実現する」というゴールを設定する（G_1）．次に，このゴールを詳細に議論するための戦略として，図２に示すアクティビティ図より，「自動運転システムのアクションから導かれる安全性要求を論じる」という戦略を設定する（S_1）．この戦略は，図２を参照するため，コンテキスト（C_1）として対象のアクティビティ図名を明記することで妥当性を確認するシステムモデルを関連付ける．

戦略（S_1）に基づき，ゴールG_2からG_6を設定する．アクターのノードに関連付けられている二重の矢印はdepend on（依存）の関係を示す．例えば，ゴールG_2「ADSは交通状況を特定できる」の達成に関して，自動運転システムはある構成システム群（CSs 01）に依存するということを示している．ここで，CSs 01は，自車両（EV），自車両のドライバ（EVD），交通インフラシステム，歩行者，周辺モビリティ，情報システムを表す．これらは，図２のアクティビティ図上で示されている情報の流れをもとにして設定する．このアクター間の依存関係は，自動運転システムにとって必要な情報がアクターCSs 01から得られる必要があることを示している．

今後，ADSによる交通状況の特定のサブ機能が詳細に定義されることにより，この構成システム群（CSs 01）は単体のアクターとして示される予定である．なお，このゴールG_2は図１のアシュアランスケースに示されるゴールG_5，G_6，G_7を実現する機能であることをコンテキストのノードで示している．一方，ゴールG_4に示されている「ADSはドライバに運転権限を移譲することができる」に関して，自動運転システムがドライバに権限を移譲するためには，自動運転システムの機能のみならず，ドライバが運転を代わることができる状態にあるのか，また適切な操作を判断できるかなど，ドライバの状態に依存することがアクターを関連付けることで示されている．このように，単体のシステムの機能だけではなく，独立して運用される他のシステムと相互に補完し合いながら動作することを検討できる．また，ゴールG_4は図１のゴールG_11を実現する機能であることをコンテキストで示している．

アクティビティ図に示される安全性要求は，図３のコンテキストのノードに示すように，全体の安全性要求として検討した結果を引き継いでいる．従って，ミッション達成のために実現すべき振る舞いがアーキテクチャの検討の中で妥当に検討できていると言える．また，アクターを用いることにより，ゴールの実現に必要な構成システムを確認することが可能となる．SoSは，構成システムが常に一定であるとは限らないという特徴を持つ．SoSの構成システム，すなわちアクターに変化がある場合にこのアシュアランスケースに立ち戻ることにより，再検討すべきゴールを見つけることができる．

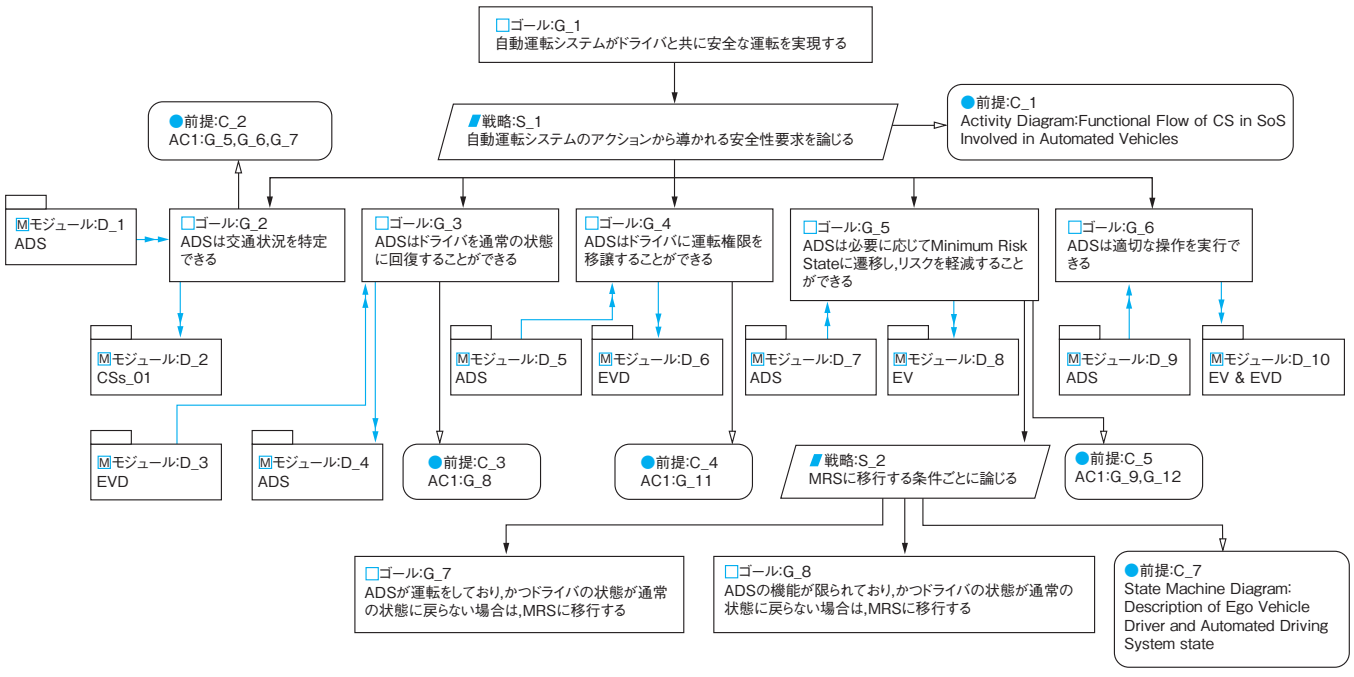


図３　アクターを構成システムとして記述したアシュアランスケース

4 CSPによるSoSアーキテクチャの検証

4.1 自動運転車を取り巻くSoSアーキテクチャのCSPモデル

3節で示した安全性要求の妥当性確認を受け，システムモデルの図で定義された振る舞いと状態の変化が安全性要求を満たすことを検証するためにCSPモデルを作成し，モデル検査を実施する．SoSの各構成システムの相互作用の結果，安全性要求が常に満たされることを検証する必要がある．CSPモデルは並行して動作するシステムの振る舞いを形式的にモデル化することができる．本論文では，自動運転車を取り巻くSoSの各構成要素を並行システムの要素とみなし，CSPモデルを構築する．CSPモデル作成のため，図2のアクティビティ図及び図4に示す状態機械図を用いる．ただし，図4の①の状態遷移はのちに示すモデル検査の結果を反映したものである．図4では，自動運転車が走行中のドライバ及び自動運転システムの状態遷移を示している．レベル3の自動運転システムでは自動運転システムとドライバとの相互作用が最も安全性に影響するという観点から，それらの相互作用に関して安全性の検証を実施する．

CSPモデルの構成を図5に示す．このCSPモデルは構成システム(CS)，自動運転システムの振る舞い(ADSProc)，ドライバの振る舞い(DriverProc)の大きな3つのプロセス群から構成されている．これらのプロセス群はCSPの並行合成を用いてSoS全体のプロセスを構成する．四角い矩形は，サブプロセスを表し，破線は各プロセスの情報交換による相互作用を表すためのチャネルを表す．また，実線の矢印は，プロセスの流れを表す．まず，自動運転システムとドライバが構成システムからの情報を認知し，その結果，それぞれの状態が変化する．ここでは，自動運転システムとドライバの状態の変化が安全性要求を常に満たすことを確認するために，それぞれの状態を変数として取り扱う．ドライバの状態の変数(evdstate)は，図4の状態の定義より，通常の運転ができる状態(normal)と眠い・注意散漫という運転をするには危険な状態(abnormal)を値として持つ．また，自動運転システムの状態(adsstate)は，状態遷移の条件より，自動運転が可能な状態(automated)と自動運転が困難な状態(limited)を値として持つ．図4上で定義される自動運転システムの状態である自動運転システムが運転をする状態(adsdriving)，ドライバが運転をする状態(evddriving)，及び最小リスク状態(Minimum Risk State)を自車両(EV)の運転の状態(evmode)として定義す

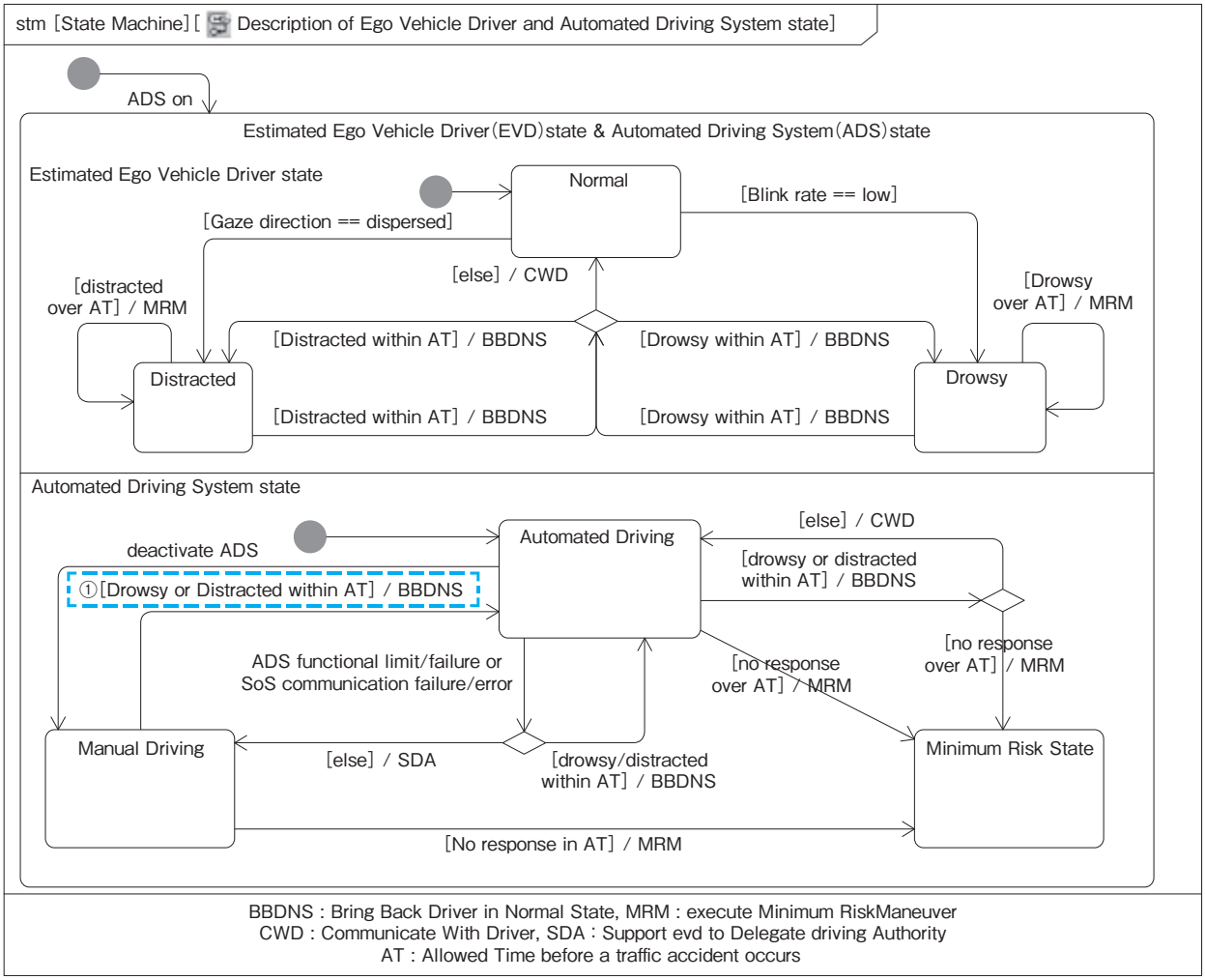


図4　自動運転車を取り巻くSystem of Systemでのドライバと自動運転システムの状態遷移

る．なお，最小リスク状態は自動運転システムが致命的な危険を避けるための状態である．

次に，図５のADSDecisionに該当するCSPモデルの自動運転システムのプロセスを表す部分を図６に示す．自動運転システムは，先に示した３種類の状態をもとに次のプロセスを選択する．これらの判断の条件文は，図２及び図４より導くことができる．例えば，自動運転システムが運転権限を持ち，かつ自動運転システムの機能では運転が困難である場合は，ドライバの状態を確認し，ドライバに権限を移譲するプロセス，またはドライバを通常の状態に戻すプロセスに移行する．ドライバが運転権限を持つ場合は，ドライバの状態を確認し，ドライバが危険な状態にある場合にドライバを通常の状態に戻すプロセスに遷移する．既に最小リスク状態に遷移している場合は，このCSPモデルでは危険を回避する自動運転システムのプロセスが続く．また，ドライバのプロセスを表すCSPモデルの記述の一部を図７に示す．図７では，ドライバが自動運転システムの振る舞いを認知することをチャネルadstoevdを介して情報を受け取るとして表現している．adstoevd を介して情報をドライバが認知したのち，ドライバ自身が運転操作をする，または自動運転システムに従うことを選択し，実行する．

4.2 モデル検査

システムモデルをもとに構築したCSPモデルに対して，検証すべき安全性要求をLTL（Linear Temporal Logic)式を用いて表現する．CSPモデルのモデル検査器であるPATはLTL式で表された条件に対して，対象のモデルが違反することがないことを検証できる[Sun2008]．LTL式の定義に反するプロセスがある場合は，PATが違反するプロセスの流れを反例として示す．この反例を解析することにより，安全性要求を満たさないシステムモデル上の対応する記述を検討する．

まず，安全性要求の妥当性を確認した図３から，検証すべき安全性要求を明確にする．図３のゴール：G_4及びG_8から求められる安全性要求をまとめると，自動運転システムが運転の継続をできない場合は，通常の状態のドライバが運転をする，または最小リスク状態に移行するという要求が導かれる．一方，自動運転システムが運転をしており，かつドライバの状態が不安全な状態の場合は，ドライバを正常な状態に戻そうとしたのち，状態が戻らないときに最小リスク状態に移行する．（図３のゴール：G_3，G_7)．これらの安全性要求を表現したLTL式の例を図８に示す．

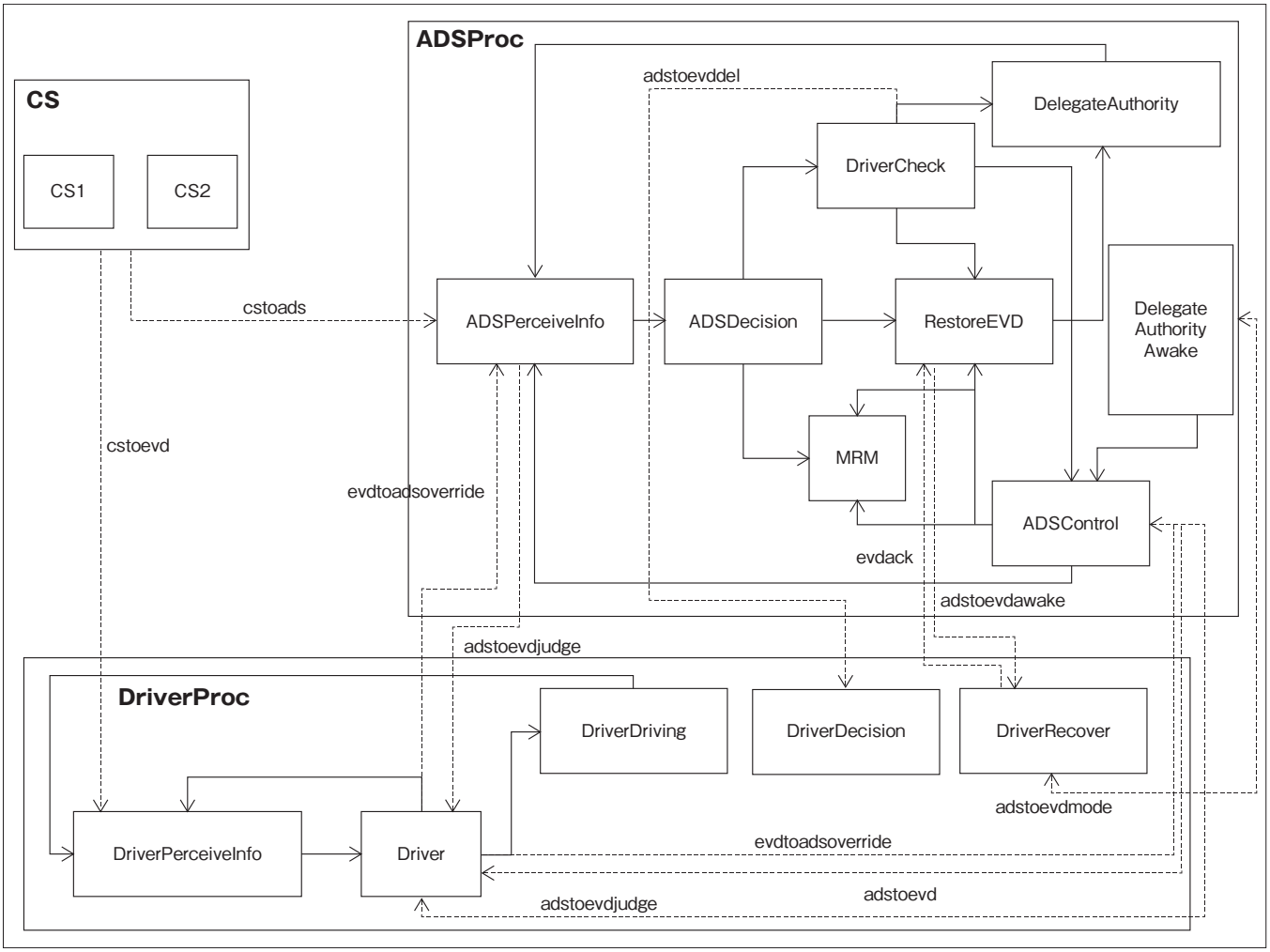


図５　CSPモデルの構成

```
ADSDecision = [evmode==evddriving && evdstate==normal && adsstate==automated](adsdecision01 -> ADSPerceiveInfo)
              [][evmode==evddriving && evdstate==normal && adsstate==limited](adsdecision02 -> ADSPerceiveInfo)
              [][evmode==evddriving && evdstate==abnormal && adsstate==automated](adsdecision03 -> RestoreEVD)
              [][evmode==evddriving && evdstate==abnormal && adsstate==limited](adsdecision04 -> MRM)
              [][evmode==adsdriving && evdstate==normal && adsstate==automated](adsdecision05 -> DriverCheck)
              [][evmode==adsdriving && evdstate==normal && adsstate==limited](adsdecision06 -> DriverCheck)
              [][evmode==adsdriving && evdstate==abnormal && adsstate==automated](adsdecision07 -> DriverCheck)
              [][evmode==adsdriving && evdstate==abnormal && adsstate==limited](adsdecision08 ->MRM)
              [][evmode==evmrs](adsdecision09 -> MRM);
```

図6　自動運転システムのプロセスを表すCSPモデルの一部

```
Driver = (adstoevd?y -> evdtoadsoverride!1 -> adstoevdjudge?z ->
          ([z==0](overriderefused -> DriverPerceiveInfo) [][z==1](driverctr1{evmode=evddriving;} -> DriverDriving)))
          [](adstoevd?y -> followads01 -> DriverPerceiveInfo)
          [](followads02 -> DriverPerceiveInfo)
          [] evdtoadsoverride!1 -> adstoevdjudge?x ->
          ([x==0](refusecmd -> DriverPerceiveInfo)[][x==1](driverctr2{evmode=evddriving;}->DriverDriving));
```

図7　ドライバのプロセスを表すCSPモデルの一部

```
#define dangerous3(evdstate == abnormal);
#define ExceptForDriver(evmode == evmrs|| evmode == adsdriving);
#assert SoS |= [](dangerous3 -> <> ExceptForDriver);
```

図8　ドライバが不安全な状態の場合に安全性要求が満たされることを検証するためのLTL式

最初にCSPモデルに対して，自動運転システムとドライバのプロセスがそれ以上進まなくなるデッドロックが起こらないことを検証し，それぞれのプロセスが予期せず停止することがないことを確認した．次に，LTL式を満たさないプロセスの遷移があるか否かを検証した．その結果，自動運転システムが運転を継続することが困難な状態で，かつドライバが不安全な状態の下で，ドライバの手動運転モードが続くパターンが検出された．

PAT上で示された反例を解析すると，まず自動運転システムの機能がほかの構成システムの変化を受け，制限された状態となる．そして，ドライバに運転権限のオーバーライドを要求し，正常な状態のドライバが要求に応答して運転権限を得る．これは，システムモデルで想定されている通り，安全な権限移譲と言える．ところが，そのほかの構成システムの影響で，ドライバの手動運転下でドライバの状態が不安全な状態となる．そして，ドライバの状態が一度は回復するが，その後もドライバが手動運転を続けている．この結果から，自動運転システムが運転を続けることが困難となり，ドライバが運転を代わったのちに危険な状態となる可能性があることが分かる．ドライバと自動運転システムが協働するという観点からは，自動運転システムがドライバの手動運転時まで自動運転システムが介入する必要性を検討する必要があると言える．ほかの安全性要求に関しては，その要求に反するプロセスは示されなかった．なお，このモデル検査の結果を受けて，ドライバの状態が不安全な場合に「Manual Driving」の状態から「Automated Driving」の状態に戻るという状態遷移が図4の状態機械図に反映されている．

5　得られた結果からの考察

3節では，システムモデルに基づいてSoSの構成システムをアクターと定義したアシュアランスケースの記述を行い，自動運転車のみならず関連する周辺システムを含めた交通システム全体の安全性要求を明確に議論することができた．とくに，どの構成システムが安全性要求に対して責任を持つべきであるかを明確に記述できることは大きな利点となる．SoSには，構成システムが時間や環境の影響により変化する特徴があり，こうした変化に対してトレーサビリティを確保した上で，関連する安全性要求の議論を再度行うことが可能となる．

次に，4節では，安全性要求が構成システム間の相互

作用によって満たされることを検証するため，システムモデルに基づきCSPモデルを作成しモデル検査を施した．SoSの構成システムはそれぞれ独立して運用されるという性質を持つことから，並行して動作するシステム要素間の相互作用を表現できるCSPモデルを用いることが有効である．また，本論文では，周辺システムからの情報に依存して自動運転システムとドライバの状態が変化することを考慮した．今後はより複雑な構成システム間の相互作用を扱うための方法を研究する必要がある．

本論文で提示したSoSの検証・妥当性確認のためのアプローチを実行するためには，システムアーキテクチャを構築し，それをシステムモデルとして記述する必要がある．それには，これらの活動に関係するメンバがシステムズエンジニアリングを理解した上で，対象とするSoSの性質を把握するために議論を重ねることが前提となる．このような体制のもとで，妥当性確認された要求に基づいてアーキテクチャの検証を行うことが重要である．とくにSoSアーキテクチャには，構成システム間の複雑な相互作用があるため，アシュアランスケースの記述やモデル検査を同時並行的に，あるいは繰り返し実施することが有効である．

なお，本研究は，大学教員４名（本務を企業に置く特任准教授１名を含む）が中心に行ったものであり，合計で約40人月を要した．これには本論文に示した内容のみならず，システムモデルの記述，過去の事故事例に基づく安全性分析などが含まれる．とくに当初は試行錯誤もあり，数多くの繰り返し検討があったことを注記しておく．

6　おわりに

本論文では，自動運転車を取り巻くSoSに対し，安全性要求の妥当性確認，及びそれに基づくアーキテクチャの検証方法を提案した．まず，システムモデルに基づき構成システムを関連付けたアシュアランスケースを記述することで安全性要求の妥当性確認を行った．次に，システムモデルに対応するCSPモデルを用い，妥当性を確認した安全性要求に対するSoSアーキテクチャの検証を行う方法を示した．ここではとくに，自動運転システムとドライバの相互作用に着目し，周辺システムからの情報による状態変化を考慮した．

SoSアーキテクチャに対するアシュアランスケースの記述では，ゴールに構成システムを関連付けるためにアクターのノードを利用し，構成システムの変化に対するトレーサビリティの確保を意図した．また，文献［西村2016］で定義したSoSアーキテクチャの自動運転システムとドライバの相互作用について検証を行ったところ，自動運転システムによる運転の継続が困難である状態で，かつ手動運転に移行したのち，ドライバが不安全な状態になる可能性があることを見出すことができた．

今後は，ドライバ特性や人とシステムのコミュニケーションなどをCSPモデルに導入した上で，モデル検査による安全性要求の検証を行いたいと考えている．

謝辞

本論文は，独立行政法人情報処理推進機構　技術本部　ソフトウェア高信頼化センター（SEC：Software Reliability Enhancement Center）が実施した「2014年度ソフトウェア工学分野の先導的研究支援事業」の支援を受けたものである．

【参考文献】

［内閣府2016］戦略的イノベーション創造プログラム（SIP）自動走行システム　研究計画，内閣府，（2016）．
［BCG2015］自動運転車市場の将来予測，ボストンコンサルティンググループ，2015，（http://www.bcg.co.jp/documents/file197533.pdf）．
［Jamshidi2011］M. Jamshidi, System of systems engineering: innovations for the twenty-first century, John Wiley & Sons, 2011.
［DeLaurentis2005］D. DeLaurentis, Understanding transportation as a system of systems design problem, presented at the 43rd AIAA Aerosp. Sci. Meet., Reno, NV, 2005.
［COMPASS］COMPASS, Available: http://www.compass-research.eu/index.html
［Shneider1999］S. Schneider, Concurrent and Real-time Systems: The CSP Approach, John Wiley & Sons, 1st Edition, 1999.
［SAE2014］SAE International, Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems, Tech. Rep. SAE J3016, January 2014.
［Friedenthal2014］S. Friedenthal, A. Moore, and R. Steiner, A practical guide to SysML: the systems modeling language, Morgan Kaufmann, Elsevier, 2014.
［西村2016］西村秀和，ユンソンギル，木下聡子，北村憲康，自動運転車を取り巻くSystem of Systems のシステムモデル構築，日本機械学会2016年度年次大会，2016.
［ユン2016］ユンソンギル，木下聡子，北村憲康，西村秀和，自動運転車を取り巻くSystem of Systems の構成システムに対する安全性要求の明確化，日本機械学会2016年度年次大会，2016.
［山本2014］山本修一郎，アシュアランスケース入門，名古屋大学，2014.
［Goodger2012］A. C. Goodger, N. H. M. Caldwell, and J. T. Knowles, What does the Assurance Case Approach deliver for Critical Information Infrastructure Protection in cybersecurity?, 7th IET International Conference on System Safety, pp. 1-6, 2012.
［Saruwatari2014］T. Saruwatari, and S. Yamamoto, D* framework creation procedure from collaboration diagram, IT CoNvergence PRActice（INPRA）, Vol. 2, No. 2, pp. 43-54, 2014.
［Kinoshita2015］S. Kinoshita, S. Yun, N. Kitamura, and H. Nishimura, Analysis of a driver and automated driving system interaction using a communicating sequential process, 2015 IEEE International Symposium on Systems Engineering（ISSE）, pp. 272-277, 2015.
［Sun2008］J. Sun, Y. Liu, and J. S. Dong, Model checking CSP revisited: Introducing a process analysis toolkit, in Leveraging Applications of Formal Methods, Verification and Validation, Springer Berlin Heidelberg, 2008, pp. 307-322.