

# MathWorks製品によるシステムズエンジニアリング 要求仕様管理

**MathWorks Japan**

**アプリケーションエンジニアリング部**

**鎌谷 祐貴**

# アジェンダ

## システムズエンジニアリングにおけるSystem Composer™とMW製品群の連携

- システムズエンジニアリングとは
  - システムズエンジニアリングのプロセス
  - システムズエンジニアリングにおけるMW製品活用
- Demo
  1. アーキテクチャと要求の紐づけ
  2. システムレベルのアーキテクチャ解析
  3. システム振る舞いの妥当性確認
- まとめ



# システムズエンジニアリングとは？

## ■ システムとは？

- 複数要素（コンポーネント）で構成されている集合体
- コンポーネントは互いに作用しあうことでシステム全体としての機能を実現

## システムズエンジニアリングとは？

- システムズエンジニアリングとは？

→“システム設計成功を実現する学際的なアプローチと手段”

- システムズエンジニアリングは、必ずしもエンジニアのものとは限りません 開発環境の外にいる者（仕様を考える企画など）を巻き込む可能性があります

# システムズエンジニアリングのプロセス

## 1. システム設計

### – 要求分析

- ステークホルダー要求を分析することで要求仕様をつくる

### – アーキテクチャ設計

- 要求仕様を実現するためのシステムの構成（アーキテクチャ）を設計する
- 各コンポーネントごとの派生要求を作成する

## 2. 設計評価・解析

- アーキテクチャが要求仕様に基づき正しく設計されているか確認する

## 3. システム統合評価

- 全体の統合評価によってステークホルダー要求を満たすかを確認する

# システムズエンジニアリングのプロセス

## 1. システム設計

### – 要求分析

- ステークホルダー要求を分析することで要求仕様をつくる

### – アーキテクチャ設計

- 要求仕様を実現するためのシステムの構成（アーキテクチャ）を設計する
- 各コンポーネントごとの派生要求を作成する

## 2. 設計評価・解析

- アーキテクチャが要求仕様に基づき正しく設計されているか確認する

## 3. システム統合評価

- 全体の統合評価によってステークホルダー要求を満たすかを確認する

# システムズエンジニアリングのプロセス ～クアッドコプタ制御開発の例～

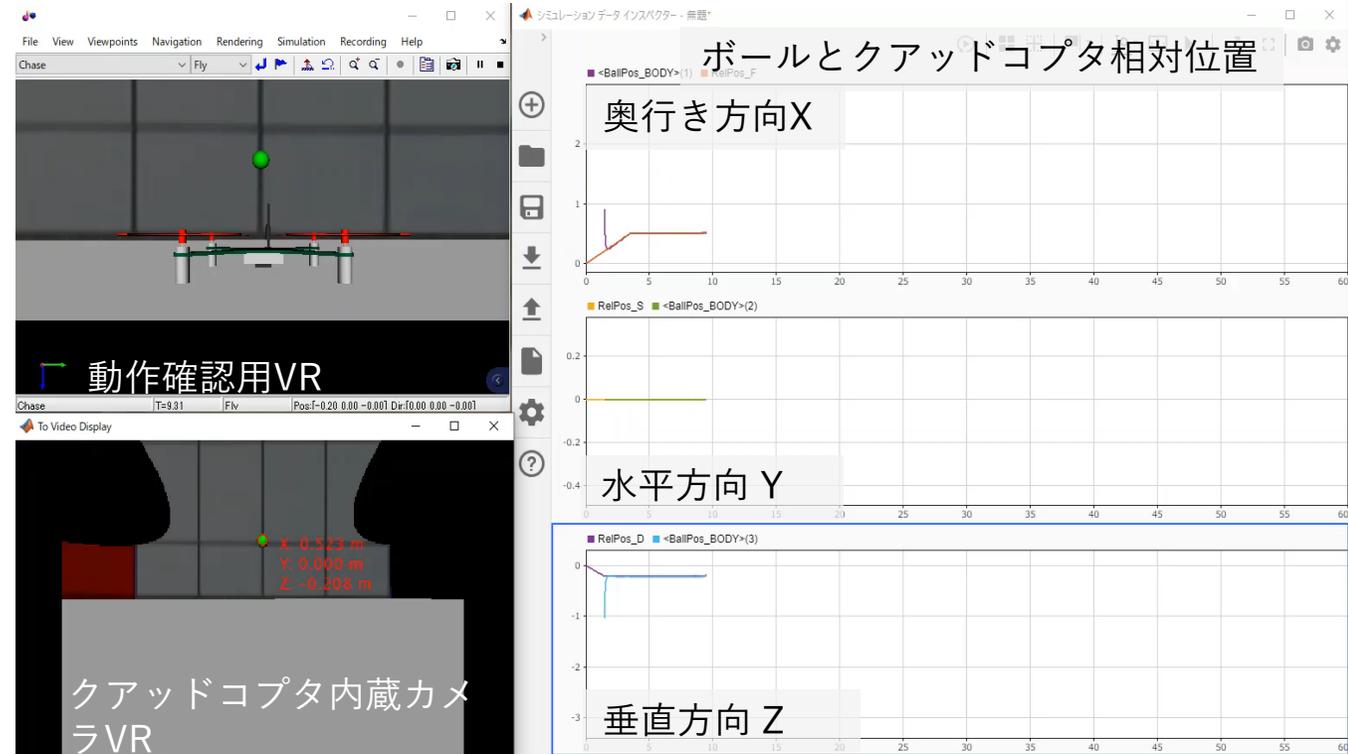
## ステークホルダ要求

緑のボールを追従するクアッドコプタ制御を可能な限り安価に構築すること

要求分析

## システム全体の要求仕様

- クアッドコプタ制御の開発
  - 緑のボールに追従
  - 対象物との相対距離誤差0.8m以内
- 部材費を\$100以下
- 評価用にドローンをVR空間にモデリング



# システムズエンジニアリングのプロセス ~クアッドコプタ制御開発の例~

## システムは複数の小さなシステムやコンポーネントの組み合わせ



### Hardware



カメラ

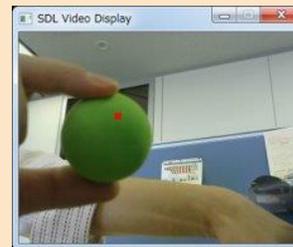


モータ

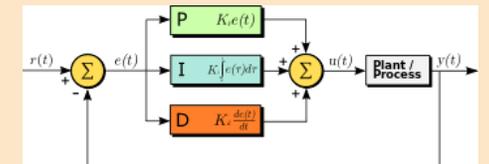


バッテリー

### Software



画像処理



モータ回転制御

# システムズエンジニアリングのプロセス ~クアッドコプタ制御開発の例~

## システムは複数の小さなシステムやコンポーネントの組み合わせ

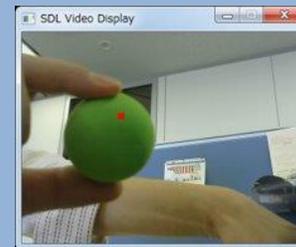


System of Systems (SoS)  
 “複数システムの組み合わせで構成されるシステム”

ターゲット  
位置推定システム



カメラ



画像処理

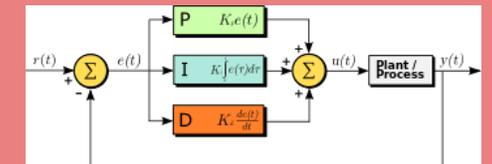
クアッドコプタ  
位置制御システム



モータ



バッテリー



モータ回転制御

クアッドコプタ制御システム

# システムズエンジニアリングのプロセス ～クアッドコプタ制御開発の例～

## システムは複数の小さなシステムやコンポーネントの組み合わせ



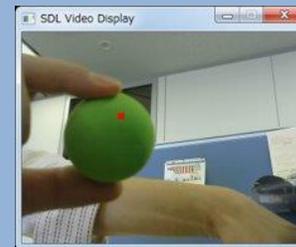
### SoS要求仕様

- クアッドコプタ制御の開発
  - 緑のボールに追従
  - 対象物との相対距離誤差 0.8m以内
- 部材費を\$100以下
- 評価用にドローンをVR空間にモデリング

### ターゲット 位置推定システム



カメラ



画像処理

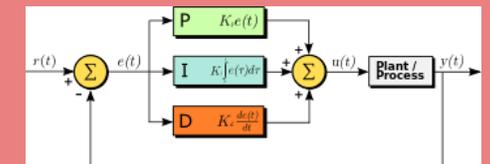
### クアッドコプタ 位置制御システム



モータ



バッテリー



モータ回転制御

## クアッドコプタ制御システム

# システムズエンジニアリングのプロセス ~クアッドコプタ制御開発の例~

## 上位要求を具体化させて各システム・コンポーネントの要求仕様に



**SoS要求仕様**

- クアッドコプタ制御の開発
  - 緑のボールに追従
  - 対象物との相対距離誤差 0.8m以内
- 部材費を\$100以下
- 評価用にドローンをVR空間にモデリング

**ターゲット位置推定システム**



**カメラ仕様**

- 解像度
- Fps
- コスト
- ...



**画像処理ソフト仕様**

- 位置推定精度
- 位置更新頻度
- ...

**クアッドコプタ位置制御システム**



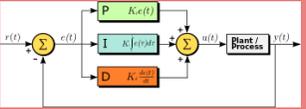
**モータ仕様**

- 回転数
- トルク
- コスト
- ...



**バッテリー仕様**

- 放電容量
- 電圧
- コスト
- ...



**制御ソフト仕様**

- 応答速度
- オーバーシュート量...

### クアッドコプタ制御システム

# システムズエンジニアリングのプロセス ～クアッドコプタ制御開発の例～

## 上位要求を満たす最適なアーキテクチャ

スペック	値
継続飛行時間	60分
ボール位置推定精度	$\pm 0.01\text{m}$
位置制御の精度	$\pm 0.05\text{m}$
コスト	\$400



スペック	値
継続飛行時間	15分
ボール位置推定精度	$\pm 0.1\text{m}$
位置制御の精度	$\pm 0.1\text{m}$
コスト	\$90



要求を実現するために最適なアーキテクチャ設計と各システム・コンポーネントへの詳細要求の落とし込みが必要

# システムズエンジニアリングにおけるツール活用の難しさ

## システムレイヤからコンポーネントレイヤまで一貫した環境が構築できない

### システム設計での課題

- ① アーキテクチャモデリングと、要求へのトレースが難しい/不可能

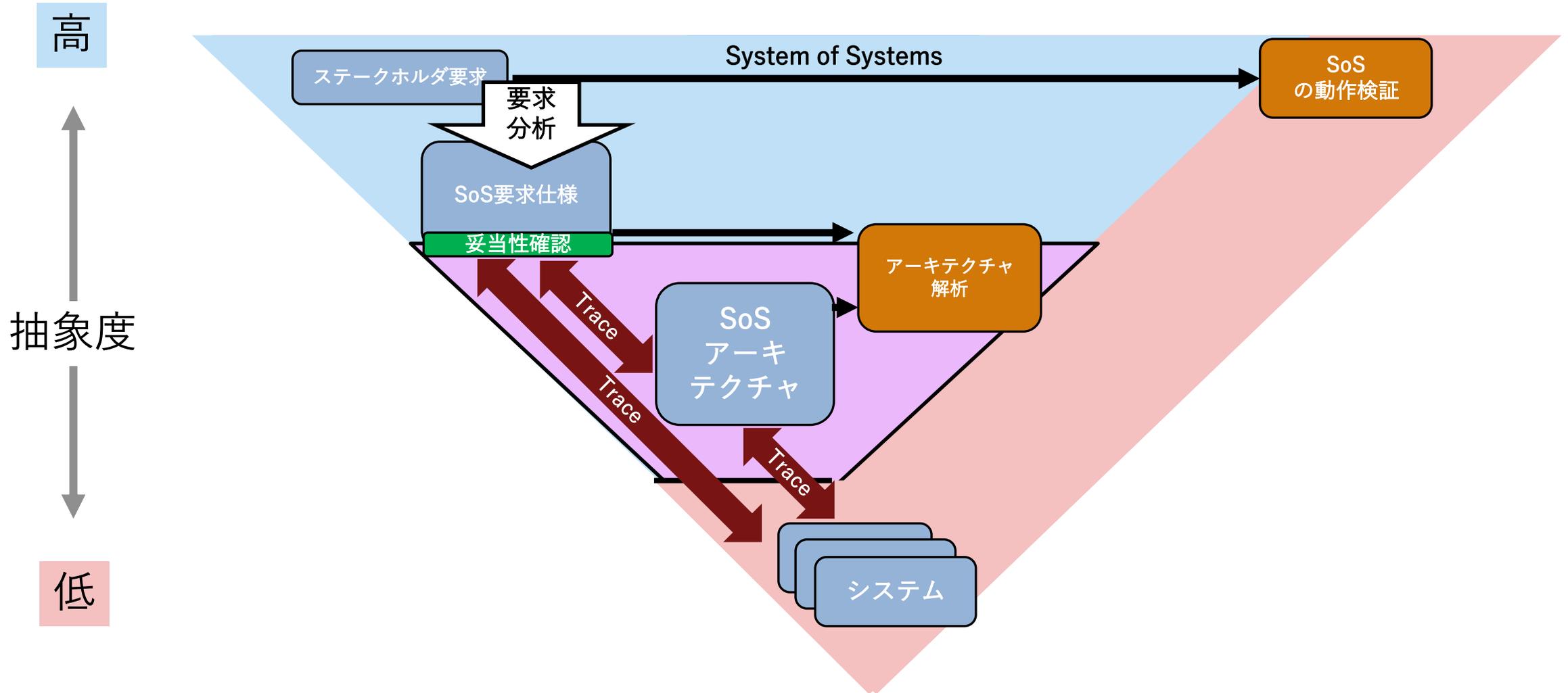
### 設計評価・解析での課題

- ② システムレベルでのアーキテクチャ検討方法が不明瞭

### システム統合評価での課題

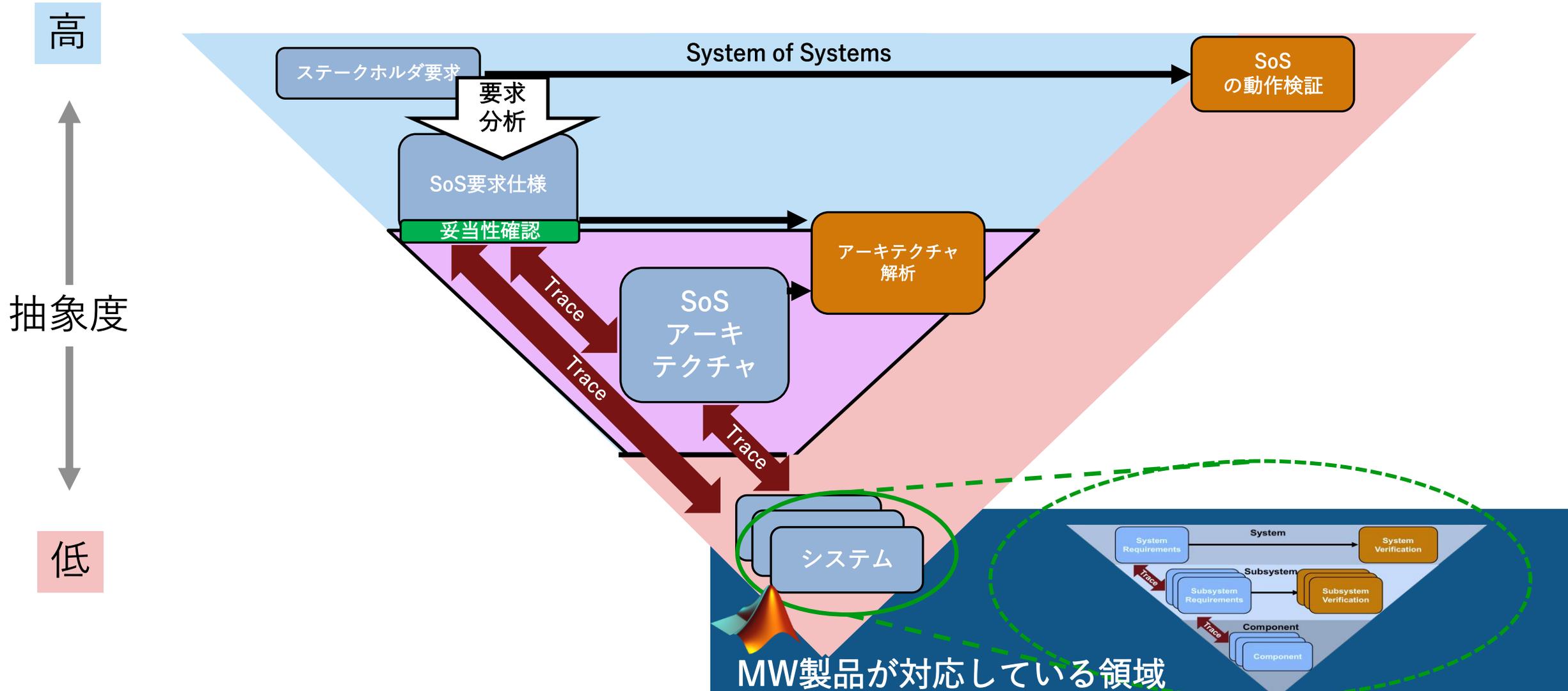
- ③ フィールドテストシミュレーション実行方法が不明瞭

# システムズエンジニアリングにおけるMW製品活用



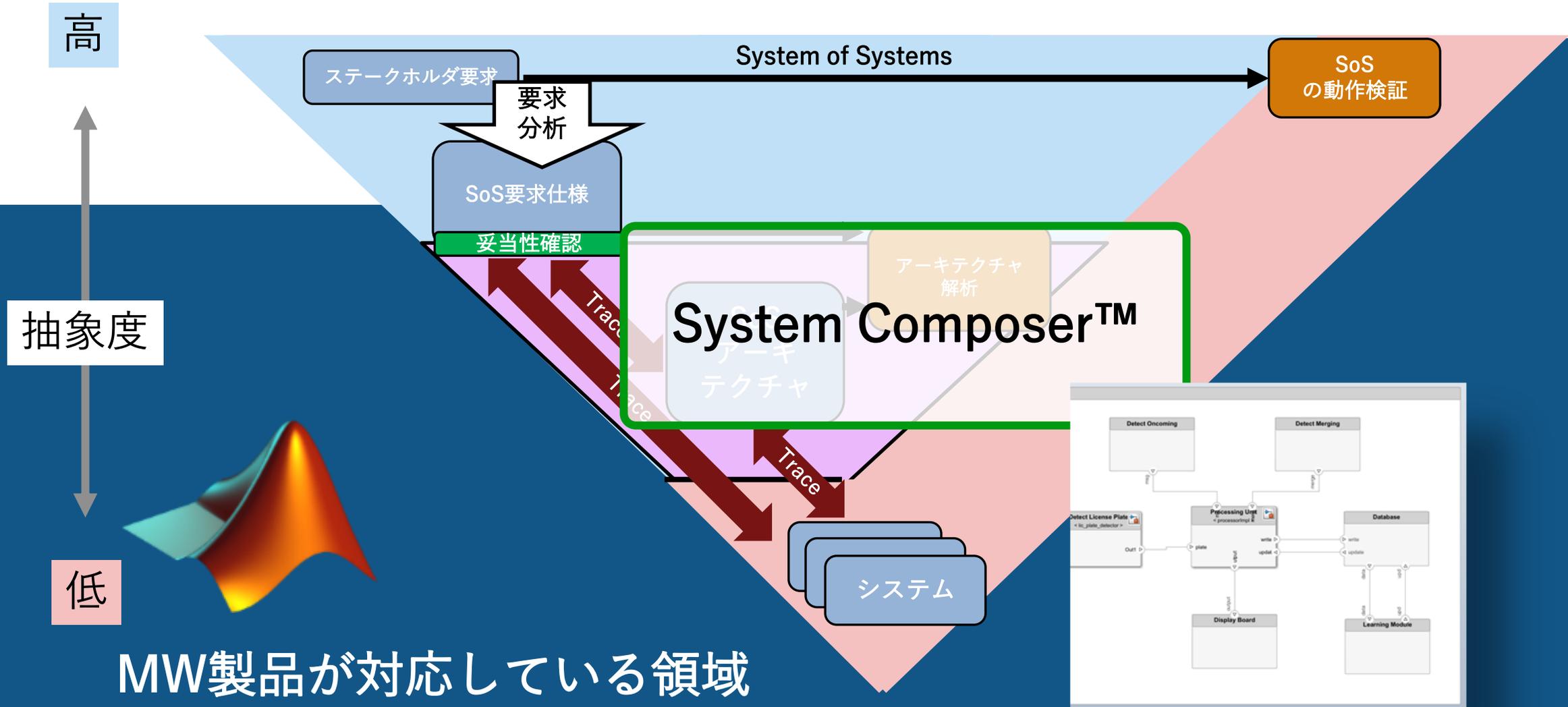
# システムズエンジニアリングにおけるMW製品活用

## 従来のSimulink製品群は抽象度の低いシステムの記述に対応

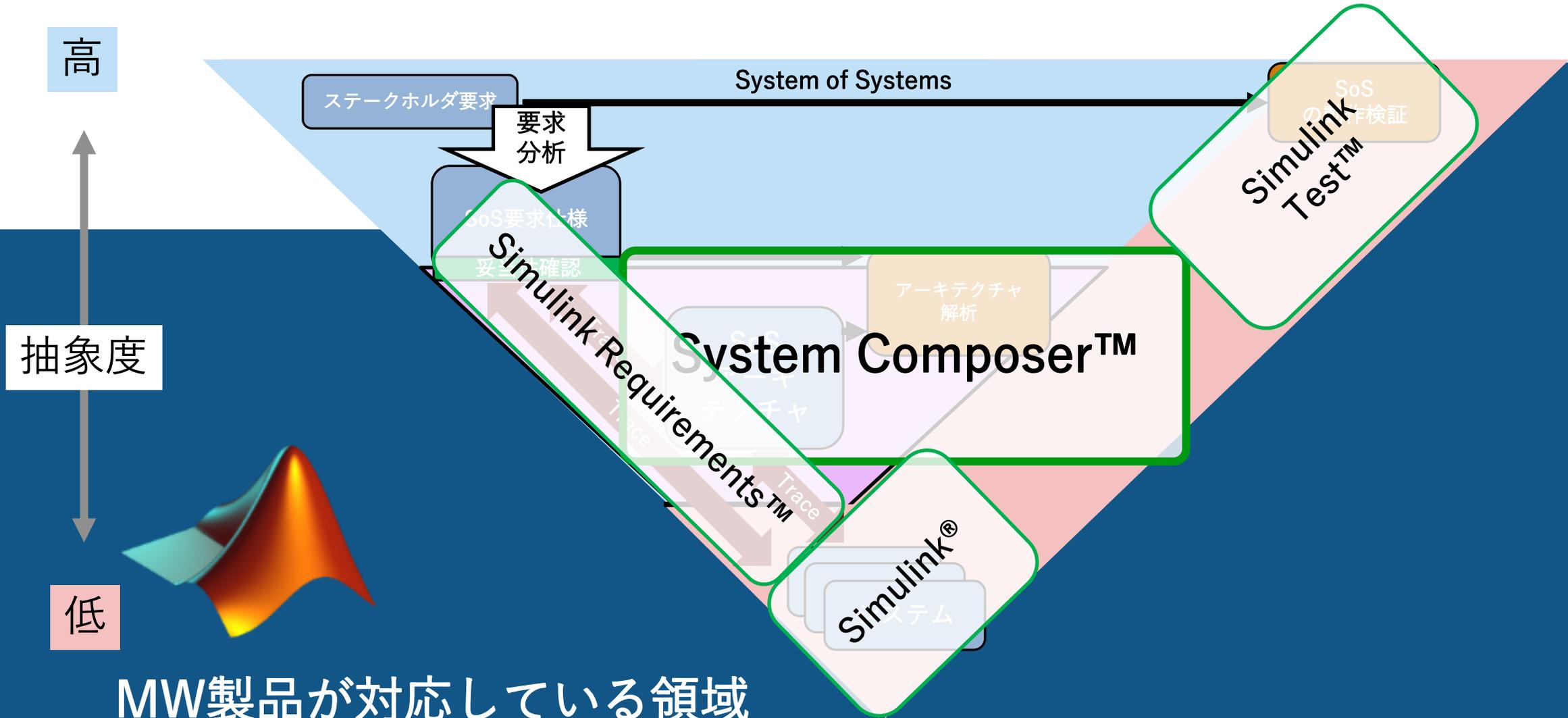


# システムズエンジニアリングにおけるMW製品活用

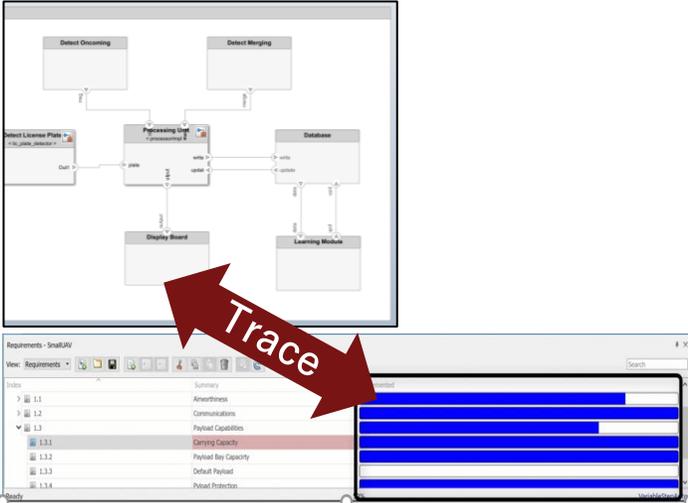
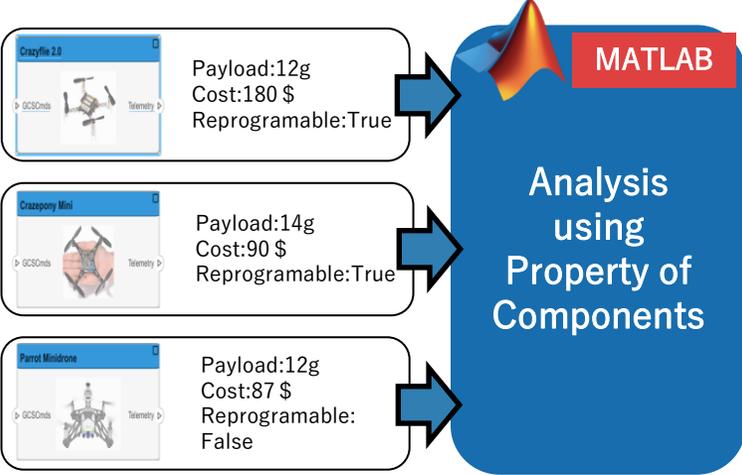
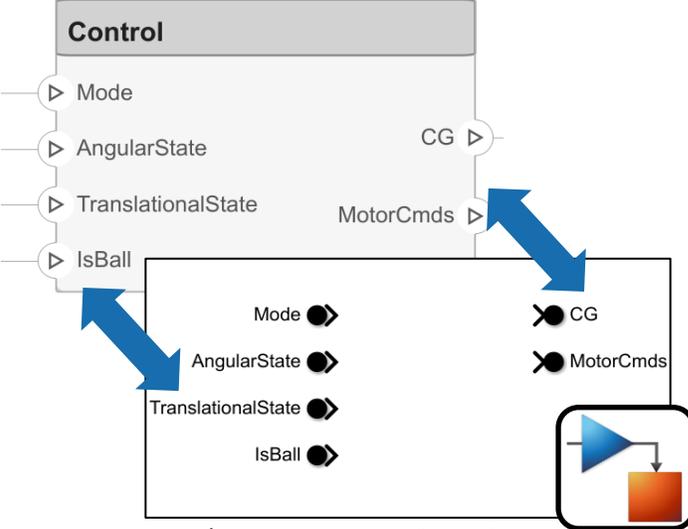
## システムズエンジニアリング領域の抽象的なモデリングが可能に



# システムズエンジニアリングにおけるMW製品活用 製品群と連携しシステムズエンジニアリング領域に対応

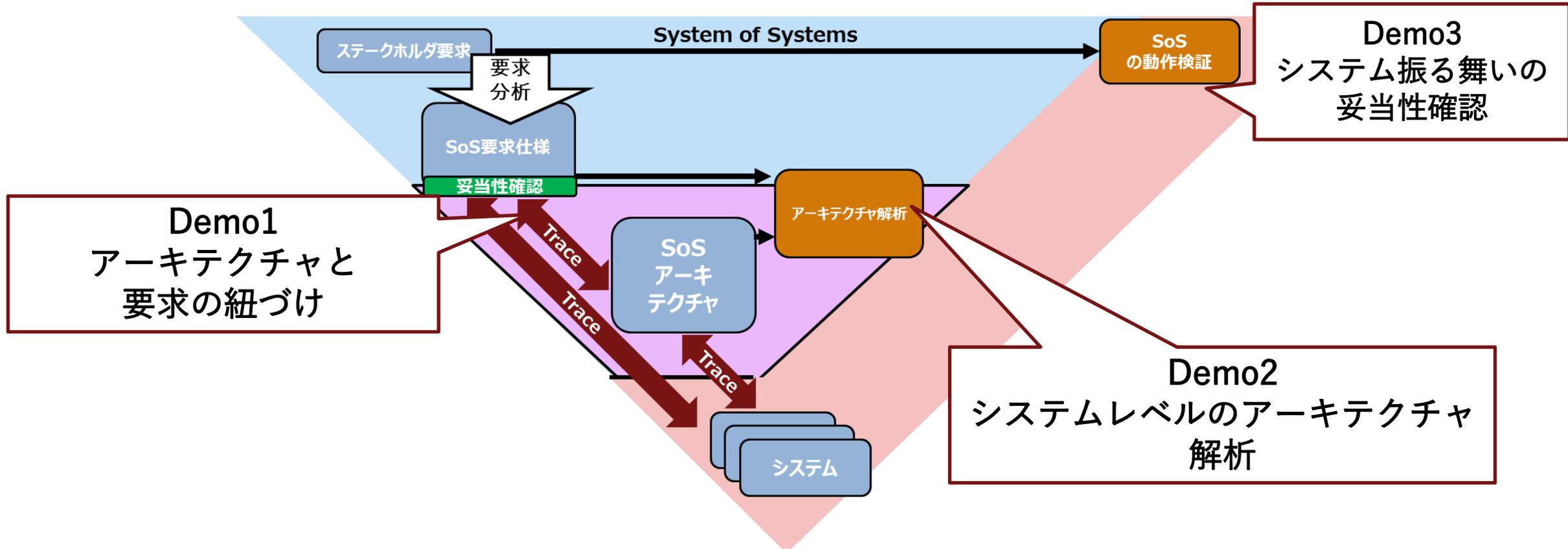


# System Composerはアーキテクチャ設計をサポートします

コンポーネントモデル作成	システムレベルでの解析	インタフェース情報を継承した Simulinkモデルとの連携
<p>システムが持つコンポーネントとインタフェースのモデリングが可能</p>	<p>コンポーネントモデルに定義した属性情報を用いた解析を MATLAB と連携して実施可能</p>	<p>インタフェースを継承した Simulink参照モデルをリンクさせ、一貫した統合シミュレーション環境を構築</p>
 <p>Simulink Requirements™ と連携すれば要件トレーサビリティ確保が可能</p>		 <p>Simulinkモデルにコンポーネントの振る舞いを記載することで動的シミュレーションを実施可能</p>

# System Composerの3つの機能を デモ動画を交えてご紹介します

コンポーネントモデル作成	システムレベルでの解析	インタフェース情報を継承した Simulinkモデルとの連携
システムを持つコンポーネントとコンポーネント間のインタフェースのモデリングが可能	コンポーネントモデルに定義した属性情報を用いた解析を MATLAB と連携して実施可能	アーキテクチャ設計で設計したインタフェース情報を継承したスケルトンモデルを生成可能
Demo1	Demo2	Demo3

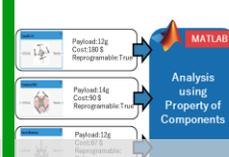
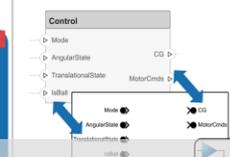


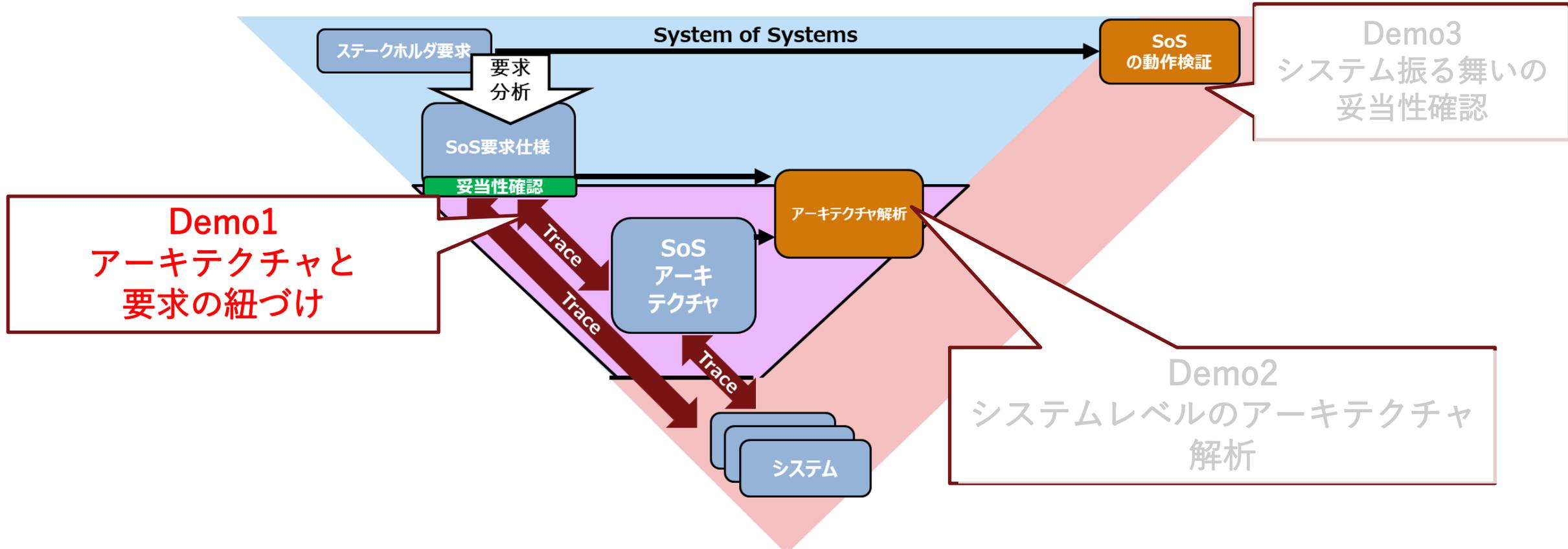
**Demo1**  
アーキテクチャと  
要求の紐づけ

**Demo3**  
システム振る舞いの  
妥当性確認

**Demo2**  
システムレベルのアーキテクチャ  
解析

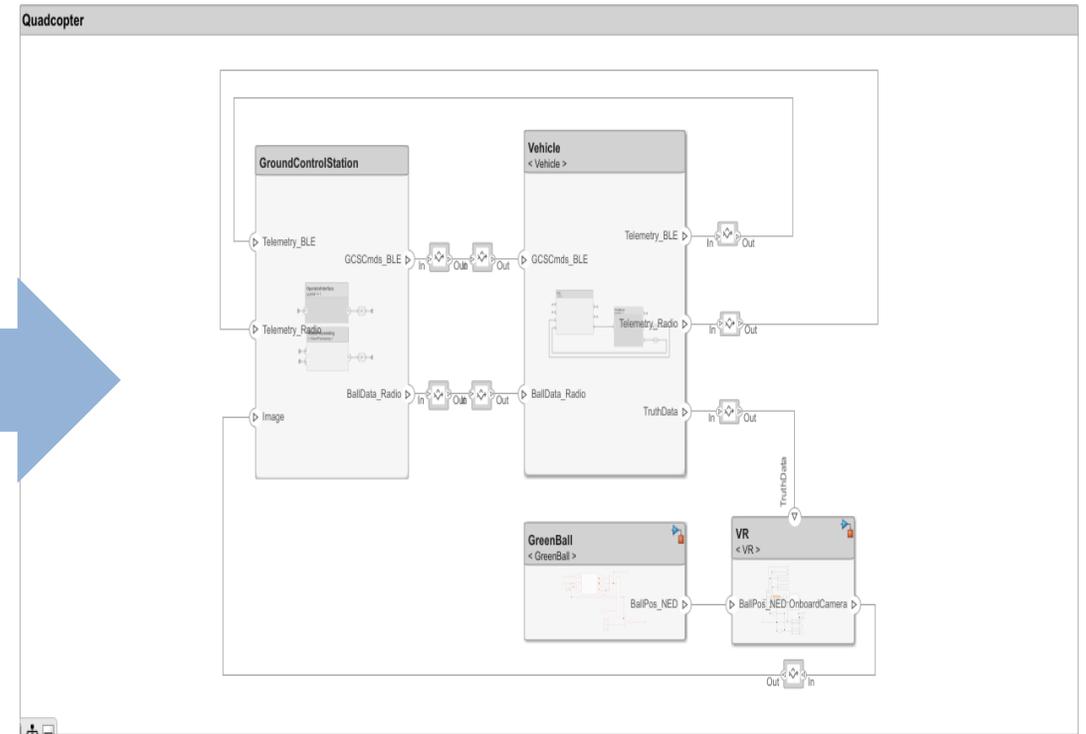
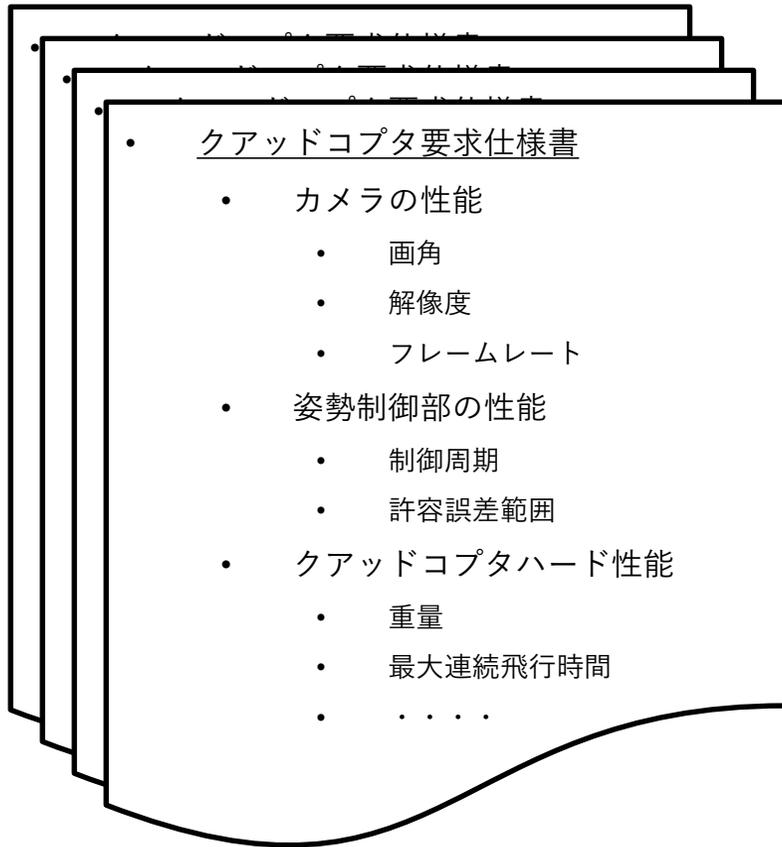
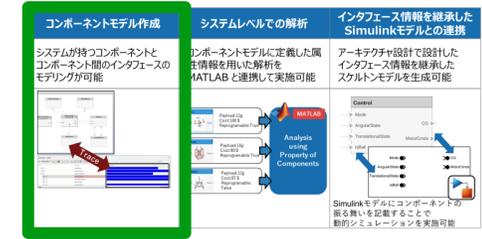
# System Composerの3つの機能を デモ動画を交えてご紹介します

コンポーネントモデル作成	システムレベルでの解析	インタフェース情報を継承した Simulinkモデルとの連携
システムを持つコンポーネントとコンポーネント間のインタフェースのモデリングが可能	コンポーネントモデルに定義した属性情報を用いた解析をMATLABと連携して実施可能	アーキテクチャ設計で設計したインタフェース情報を継承したスケルトンモデルを生成可能
		
Demo1	Demo2	Demo3
		動的シミュレーションを実施可能



# Demo1 : アーキテクチャと要求の紐づけ

## アーキテクチャと要求の対応付けをどのように管理すればよい？

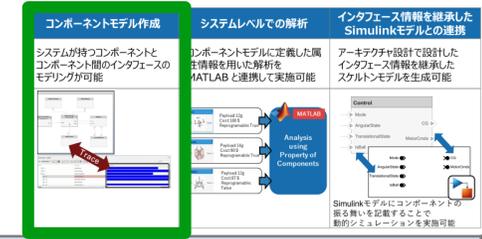


複数の要求仕様書・派生要求仕様書

アーキテクチャモデル

# Demo1 : アーキテクチャと要求の紐づけ

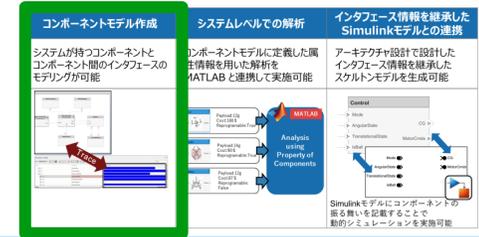
## Simulink Requirements™で要件の作成、管理、トレースを実現



要件エディタ	要件パースペクティブ	要件トレーサビリティ
<p>Simulink上で外部要件と内部要件の管理と分析</p>	<p>要件をドラッグ&amp;ドロップしてリンクを作成</p>	<p>要件と設計、コード、テスト間で双方向にトレーサビリティを確保</p>
<p>外部要件</p> <ul style="list-style-type: none"> <li>Word</li> <li>Excel</li> <li>DOORS</li> </ul> <p>インポート</p>	<p>ドラッグ &amp; ドロップ</p>	<p>要件</p> <p>テスト</p> <p>設計</p> <p>コード</p>

# Demo1 : アーキテクチャと要求の紐づけ

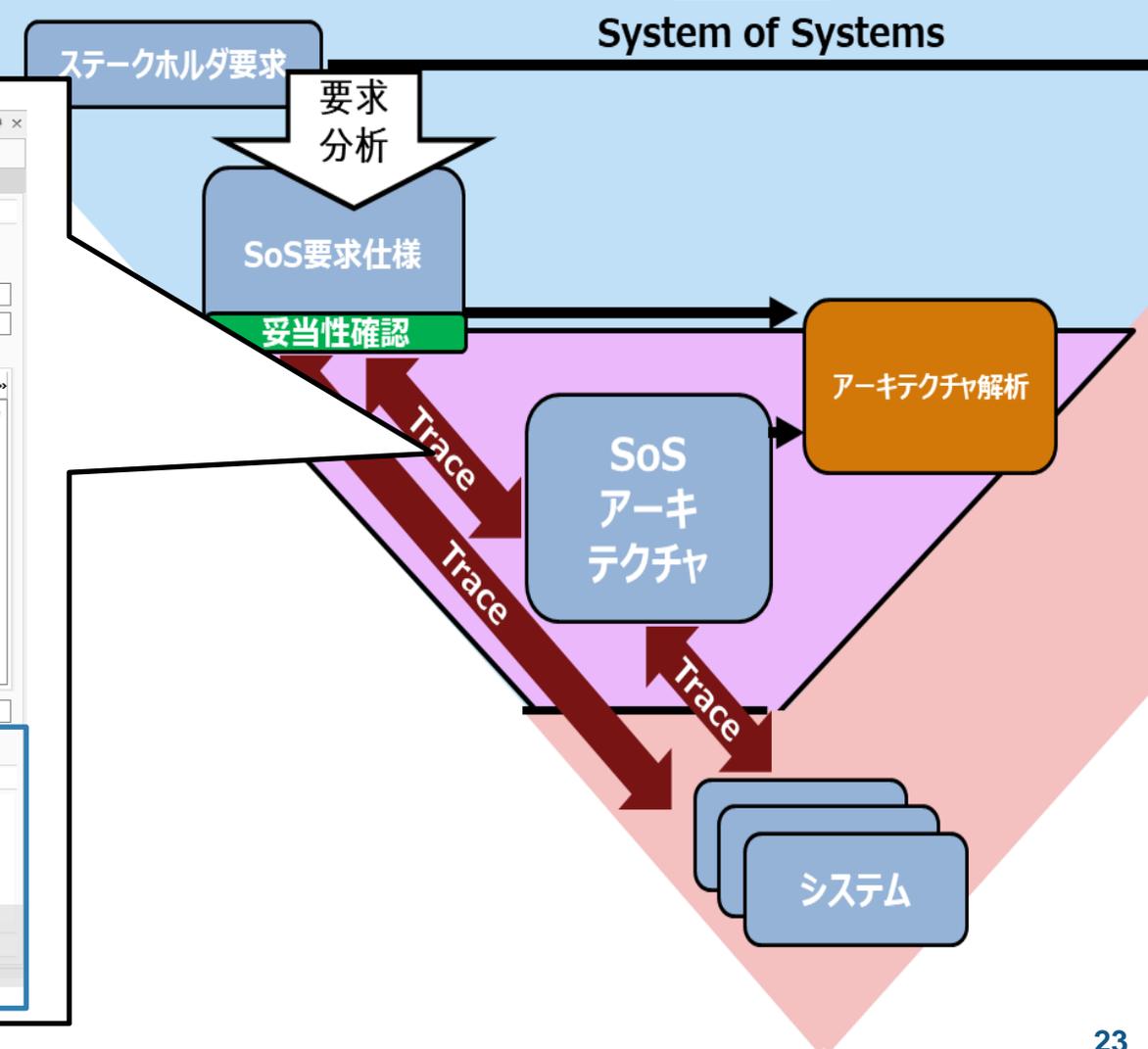
## Simulinkモデルと同じ感覚で要求とアーキテクチャの紐づけ可能



### アーキテクチャコンポジション

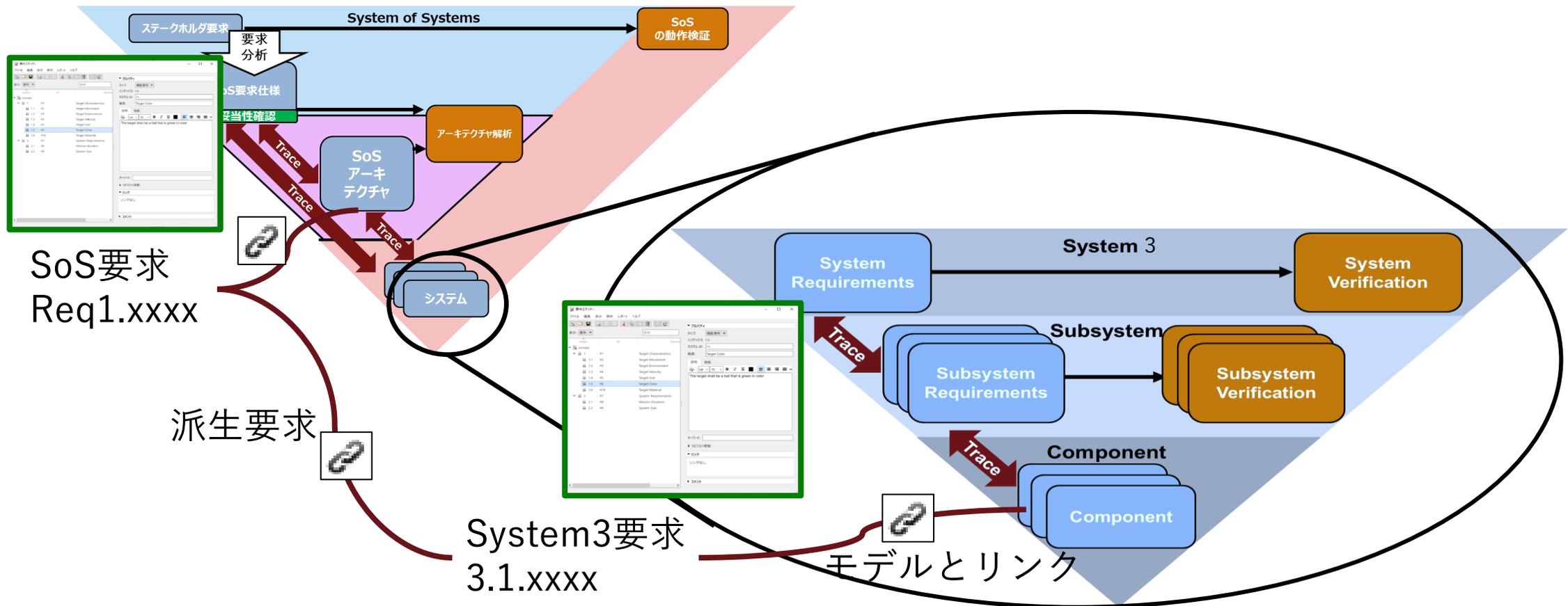
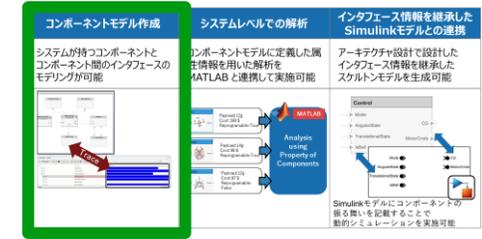
#### 要求ビューワー

Index	ID	Summary
conops		
1	#1	Target Characteristics
1.1	#2	Target Movement
1.2	#3	Target Environment
	#4	Target Velocity



# Demo1: アーキテクチャと要求の紐づけ

要求、アーキテクチャおよび設計間のトレーサビリティを可能にするための一貫した情報の流れ (デジタルスレッド) の確立を可能にします。



# Demo1 : アーキテクチャと要求の紐づけ (デモ動画)

## Simulink Requirements™で要求の管理、トレーズを実現



要件エディター

ファイル 編集 表示 解析 レポート ヘルプ

表示: 要件 検索

Index	ID	Summary
✓ conops*		
1	#1	Target Characteristics
1.3	#4	Target Velocity
1.4	#5	Target Size
1.1	#2	Target Movement
1.6	#10	Target Material
1.2	#3	Target Environment
1.5	#6	Target Color
2	#7	System Requirements
2.2	#9	System Size
2.1	#8	Mission Duration
✓ quadcopter		
> 2	#2	Power System
> 5	#5	Payload
> 3	#3	Mass Properties & Geometry
> 4	#4	Flight Control System
> 1	#1	Aircraft Performance

▼ プロパティ

ファイルパス: C:\¥LocalWS¥Project¥SystemComposerMO¥00\_MW\_GitLab¥quad\_copter\_...

リビジョン: 5

作成者: manthony

作成日: 13-4月-2019 01:13:04

更新者: cbrooks

変更日: 11-12月-2019 06:57:47

説明:

▶ カスタム属性レジストリ

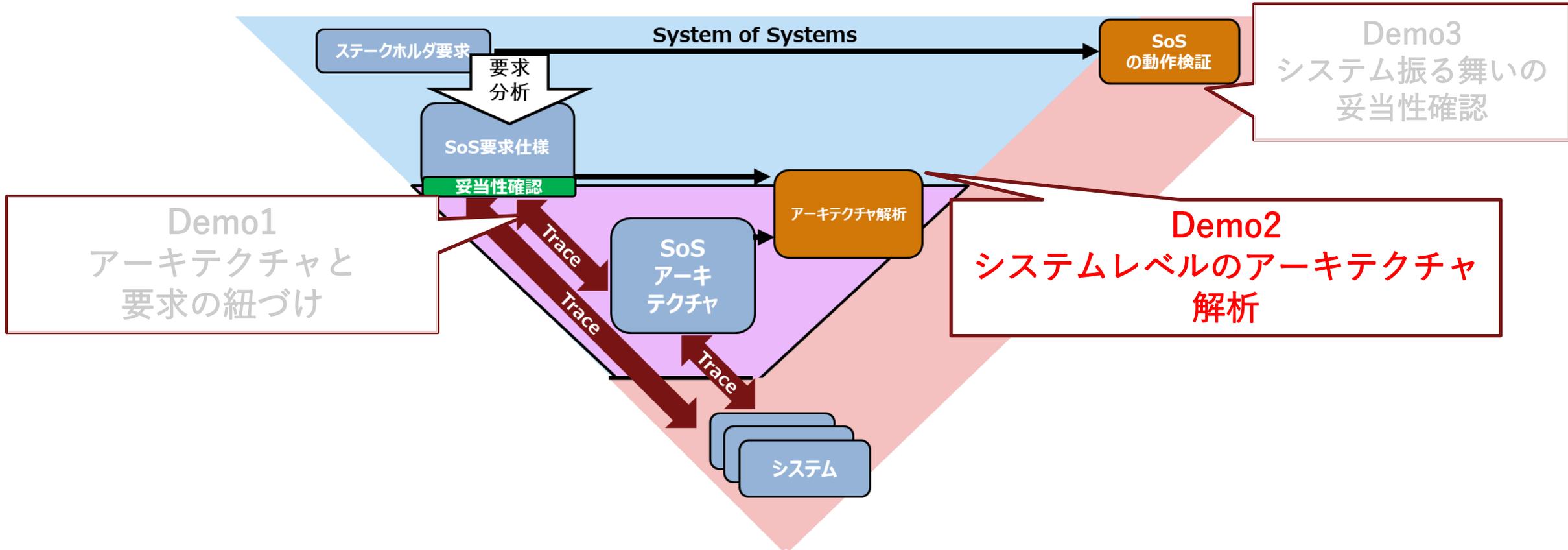
Simulink Requirementsの機能

①要件エディタ

Doors,IFReq,MS Word / Excelから取り込んだ要件情報を管理・編集できます

# System Composerの3つの機能を デモ動画を交えてご紹介します

コンポーネントモデル作成	システムレベルでの解析	インタフェース情報を継承した Simulinkモデルとの連携
システムを持つコンポーネントとコンポーネント間のインタフェースのモデリングが可能	コンポーネントモデルに定義した属性情報を用いた解析をMATLABと連携して実施可能	アーキテクチャ設計で設計したインタフェース情報を継承したスケルトンモデルを生成可能
Demo1	Demo2	Demo3
		動的シミュレーションを実施可能

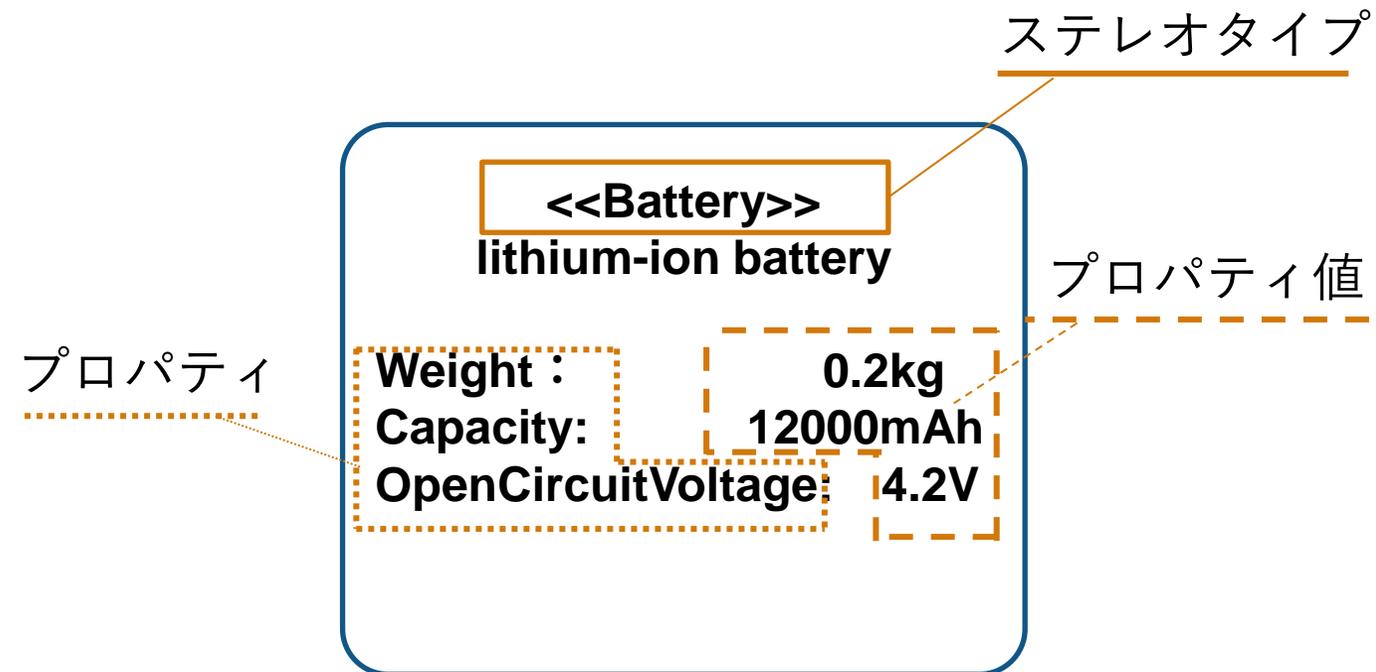


# Demo2:システムレベルのアーキテクチャ解析 ステレオタイプ・プロパティとは？



- ステレオタイプ
  - 特定用途のために拡張して定義したモデル要素・情報をまとめたもの
- プロパティ
  - ステレオタイプに付随するモデル情報のこと
  - プロパティに割り当てられた具体的な値をプロパティ値・プロパティ情報と呼ぶ

## ステレオタイプ例



## Demo2:システムレベルのアーキテクチャ解析 どのような場合にシステムレベル解析を用いるのか？



- 本システムで使用するドローンのアーキテクチャを3つに絞りました

### <<QuadCopterPhysicalProperties>>



Payload:12g  
Cost:180 \$  
Reprogramable:True

### <<QuadCopterPhysicalProperties>>



Payload:14g  
Cost:90 \$  
Reprogramable:True

### <<QuadCopterPhysicalProperties>>



Payload:12g  
Cost:87 \$  
Reprogramable:False

選定基準：リプログラミング可能 ∩ ペイロード量/コスト比最大

# Demo2: システムレベルのアーキテクチャ解析

System ComposerのAnalysis Model機能を用いると  
ステレオタイプのプロパティを演算の変数として利用可能



**Crazyflie 2.0**

Payload: 12g  
Cost: 180 \$  
Reprogrammable: True

**Crazepony Mini**

Payload: 14g  
Cost: 90 \$  
Reprogrammable: True

**Parrot Minidrone**

Payload: 12g  
Cost: 87 \$  
Reprogrammable: False

Analysis Model

**Instantiate Architecture Model**

Description

Create an instance model from this architecture model by flattening out all referenced models and their components. Such an instance model may be used for system-level analysis expressed as MATLAB functions.

Step 1: Select Stereotypes

Select the stereotypes to make available on the instance model

- QuadcopterPhysicalProperties
  - AirVehicle
  - Battery
  - HW\_Implementation
  - Payload
  - PayloadComponent
  - Target
- Strict Mode

Step 2: Configure Analysis

Function

Analysis function:

Function arguments (comma-separated):

`>> QuadChoice_PayloadCost(instance)`

Model Iteration

Iteration Order: Pre-order

Instance Model Properties

Name:

Normalize Units

Cancel Instantiate

# Demo2: システムレベルのアーキテクチャ解析

選定基準をMATLAB言語で数式として定義

ToolBoxを活用し様々な演算をプロパティ値をベースにも実施できます



**Crazyflie 2.0**

GCSCmds Telemetry

Payload: 12g  
Cost: 180 \$  
Reprogrammable: True

**Crazepony Mini**

GCSCmds Telemetry

Payload: 14g  
Cost: 90 \$  
Reprogrammable: True

**Parrot Minidrone**

GCSCmds Telemetry

Payload: 12g  
Cost: 87 \$  
Reprogrammable: False

**Analysis Model**

Instantiate Architecture Model

Description  
Create an instance model from this architecture model by flattening out all referenced models and their components. Such an instance model may be used for system-level analysis expressed as MATLAB functions.

Step 1: Select Stereotypes

Select the stereotypes to make available on the instance model

- QuadcopterPhysicalProperties
  - AirVehicle
  - Battery
  - HW\_Implementation
  - Payload
  - PayloadComponent
  - Target

Strict Mode

Step 2: Configure Analysis

Function

Analysis function:  
`QuadChoice_PayloadCost`

Function arguments (comma-separated):  
`>> QuadChoice_PayloadCost(instance)`

Model Iteration  
Iteration Order: Pre-order

Instance Model Properties  
Name: QuadArch  
 Normalize Units

Cancel Instantiate

**QuadChoice\_PayloadCost.m**

PayloadCost =  
PayloadCapacity / Cost  
\*Reprogrammable

# Demo2: システムレベルのアーキテクチャ解析

選定基準をMATLAB言語で数式として定義

ToolBoxを活用し様々な演算をプロパティ値をベースにも実施できます



Crazyflie 2.0

GCSCmds Telemetry

Payload: 12g  
Cost: 180 \$  
Reprogrammable: True

Crazepony Mini

GCSCmds Telemetry

Payload: 14g  
Cost: 90 \$  
Reprogrammable: True

Parrot Minidrone

GCSCmds Telemetry

Payload: 12g  
Cost: 87 \$  
Reprogrammable: False

Analysis Model

Instantiate Architecture Model

Description

Create an instance model from this architecture model by flattening out all referenced models and their components. Such an instance model may be used for system-level analysis expressed as MATLAB functions.

Step 1: Select Stereotypes

Select the stereotypes to make available on the instance model

- QuadcopterPhysicalProperties
  - AirVehicle
  - Battery
  - HW\_Implementation
  - Payload
  - PayloadComponent
  - Target

Strict Mode

Don't see your profile? [Profile Editor ...](#)

Step 2: Configure Analysis

Function

Analysis function:  
QuadChoice\_PayloadCost

Function arguments (comma-separated):  
>> QuadChoice\_PayloadCost(instance)

Model Iteration

Iteration Order: Pre-order

Instance Model Properties

Name: QuadArch

Normalize Units

Cancel Instantiate

QuadChoice\_PayloadCost.m

**PayloadCost = PayloadCapacity / Cost \* Reprogrammable**

数学および最適化

- Curve Fitting Toolbox
- Optimization Toolbox
- Global Optimization Toolbox
- Symbolic Math Toolbox
- Mapping Toolbox
- Partial Differential Equation Toolbox

AI、データサイエンス、統計

- Statistics and Machine Learning Toolbox
- Deep Learning Toolbox
- Reinforcement Learning Toolbox
- Deep Learning HDL Toolbox
- Text Analytics Toolbox
- Predictive Maintenance Toolbox

# Demo2: システムレベルのアーキテクチャ解析 (デモ動画)

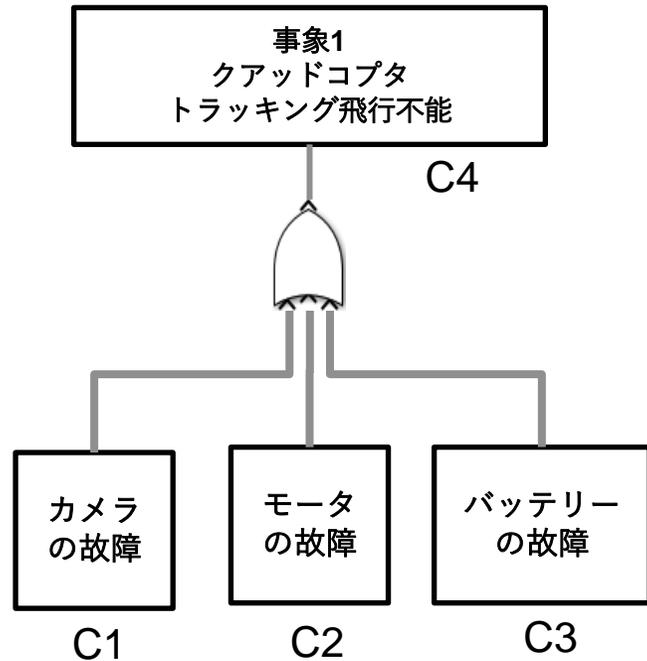
選定基準をMATLAB言語でプロパティ値を用いた  
数式として定義しAnalysis Modelに登録することで演算が可能



システムのトレードオフ検討  
1. モデルにステレオタイプを適応し、プロパティ値を入力する

# Demo2: システムレベルのアーキテクチャ解析

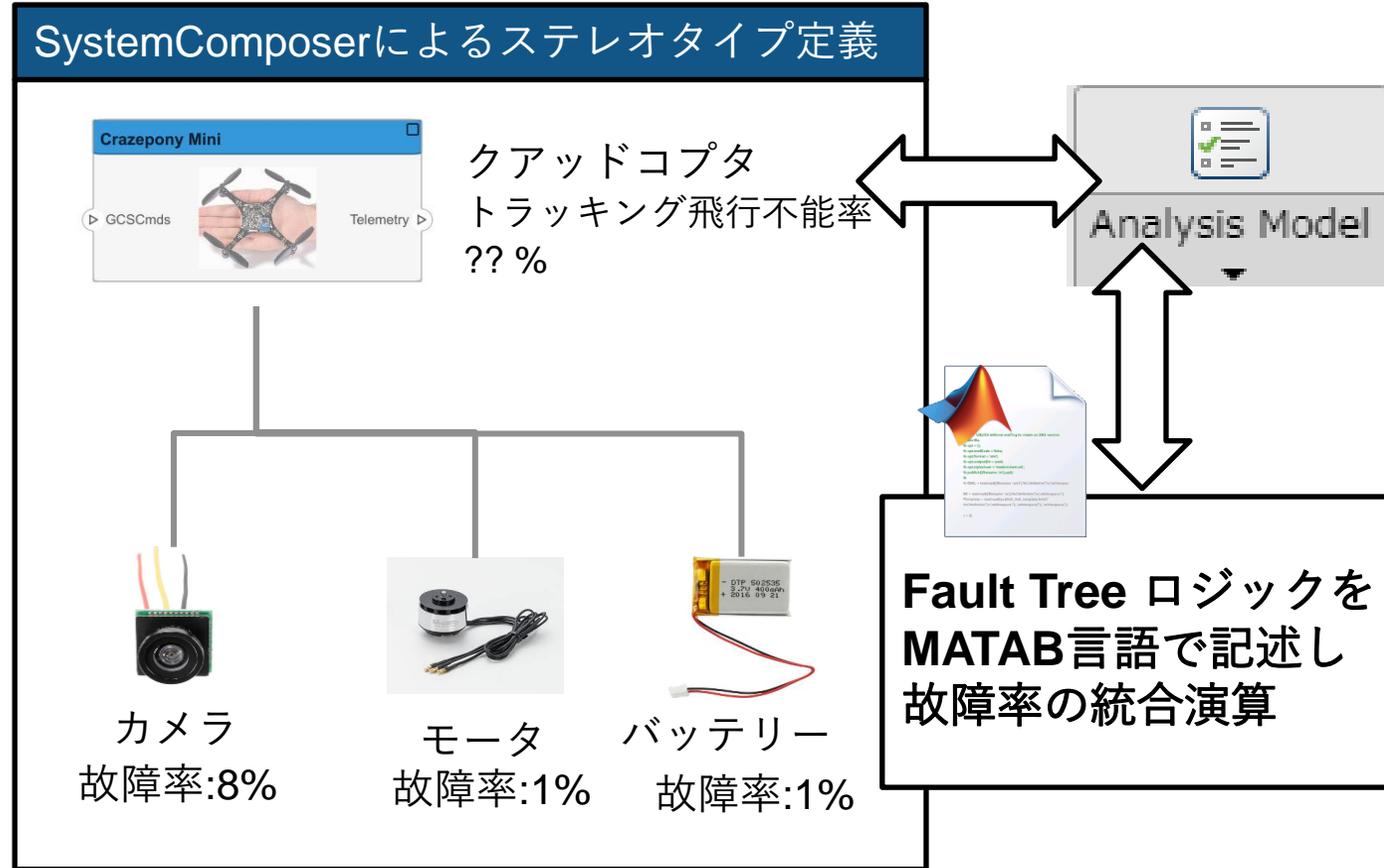
## 故障率をプロパティとして持たせればFTAも可能です



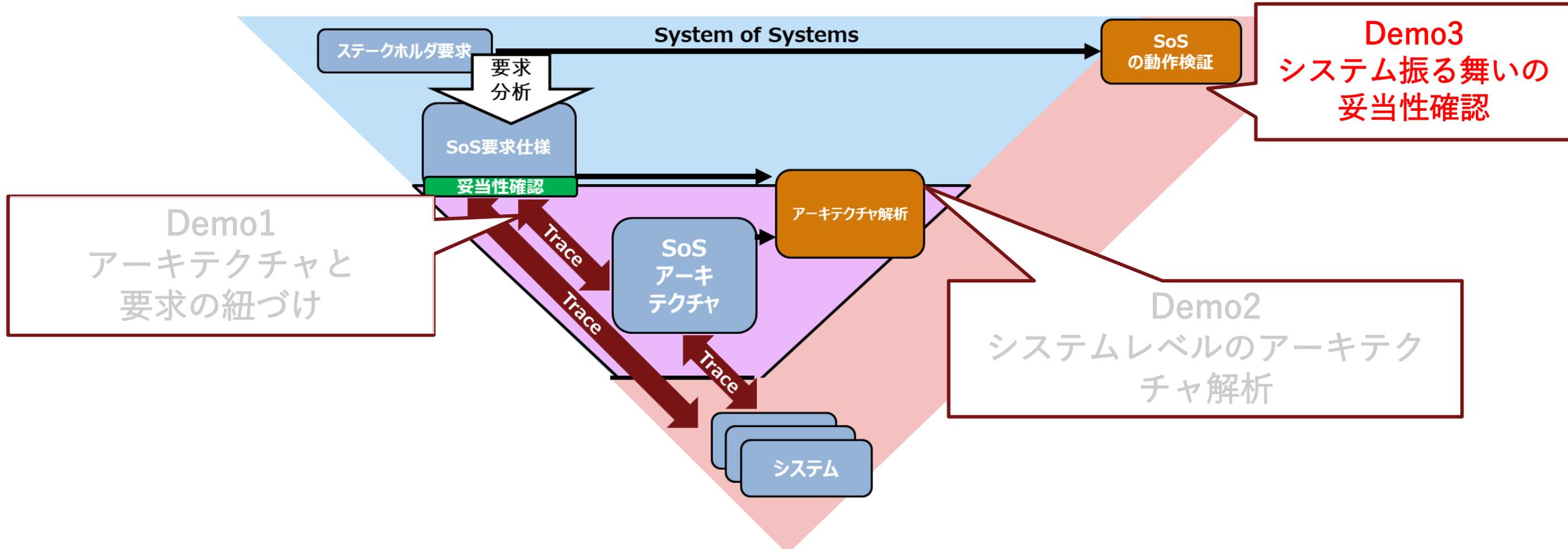
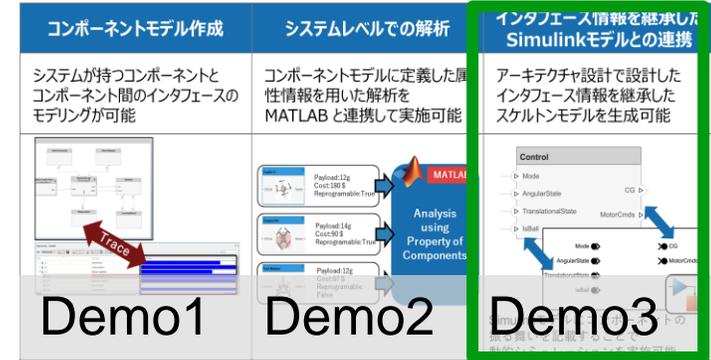
FT図一例

各事象発生率

- C1 = 8%
- C2 = 1%
- C3 = 1%
- C4 = ??

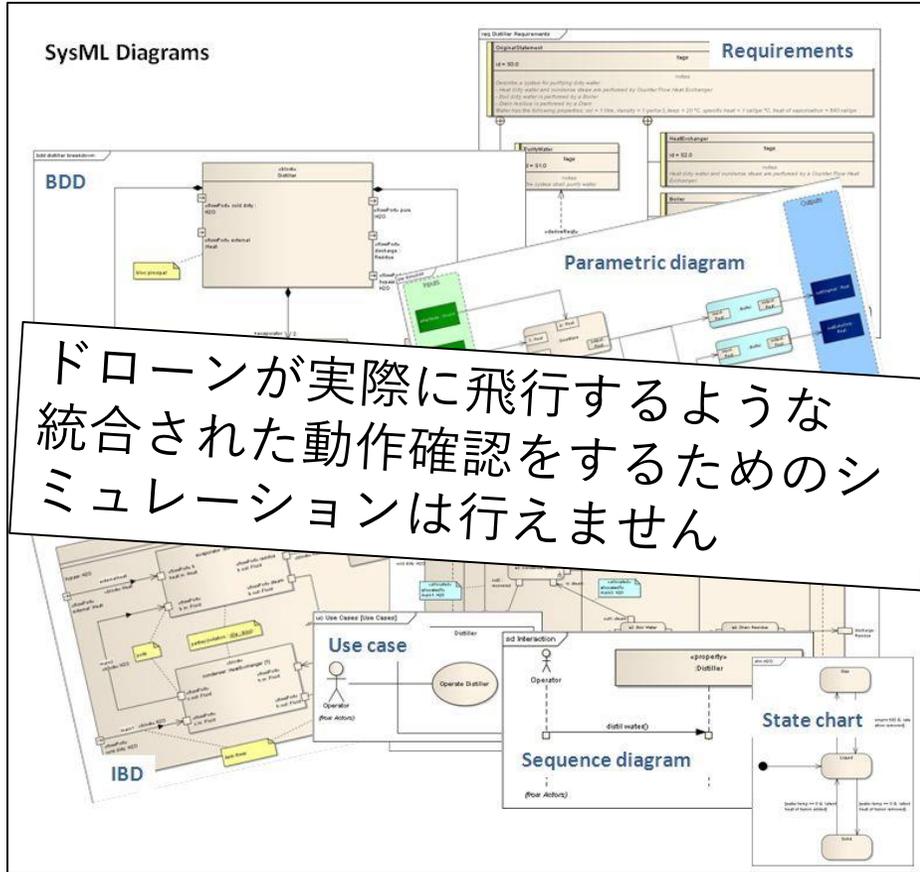
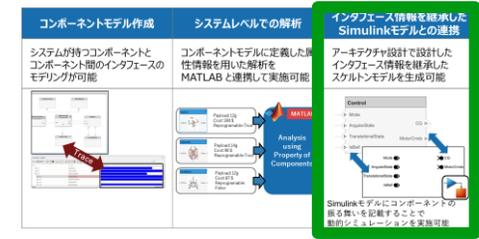


# System Composerの3つの機能を デモ動画を交えてご紹介します



# Demo3: システム振る舞いの妥当性確認

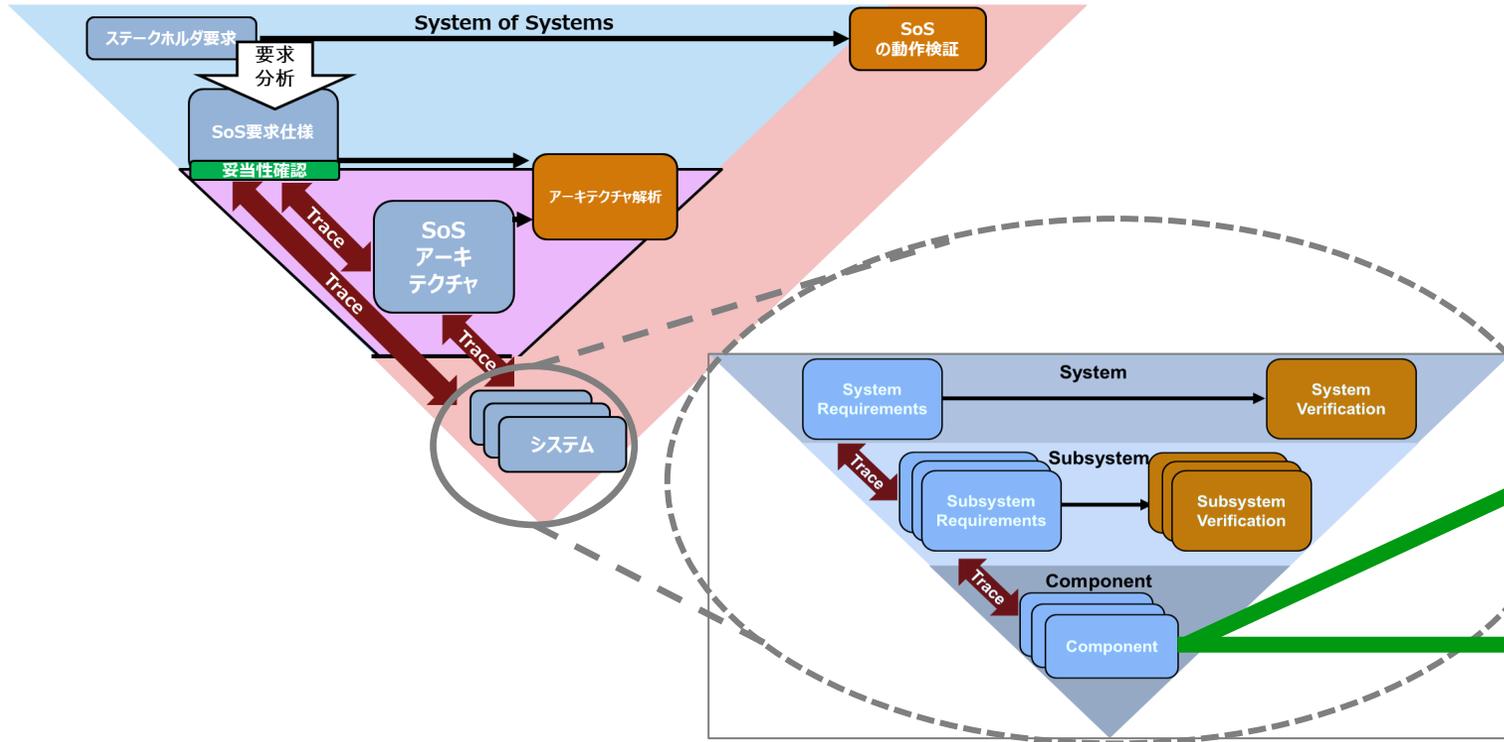
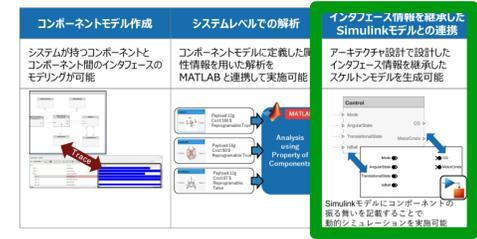
## ステークホルダ要求、SoS要求仕様達成を検証するには？



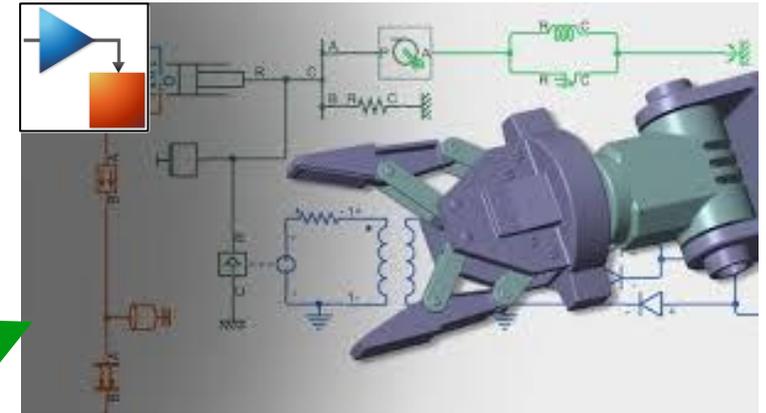
- システムズエンジニアリングで多く利用されるUML/SysMLには様々なダイアグラムが存在しますが、これらはシステムの振る舞いのある視点で切り出して表現するものです
- ステークホルダ要求、SoS要求仕様が達成されているかどうかは複数の視点での検討結果を統合して判断が必要になります

# Demo3: システム振る舞いの妥当性確認

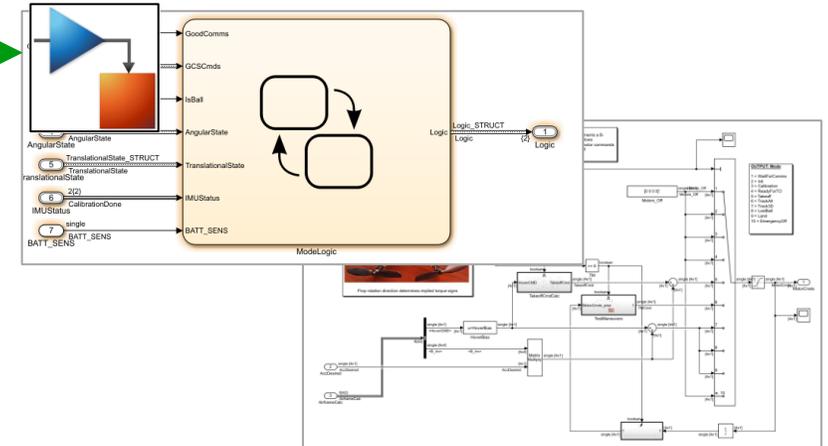
## System ComposerモデルにSimulink参照モデルを紐づけ



### Physical Device



### Software

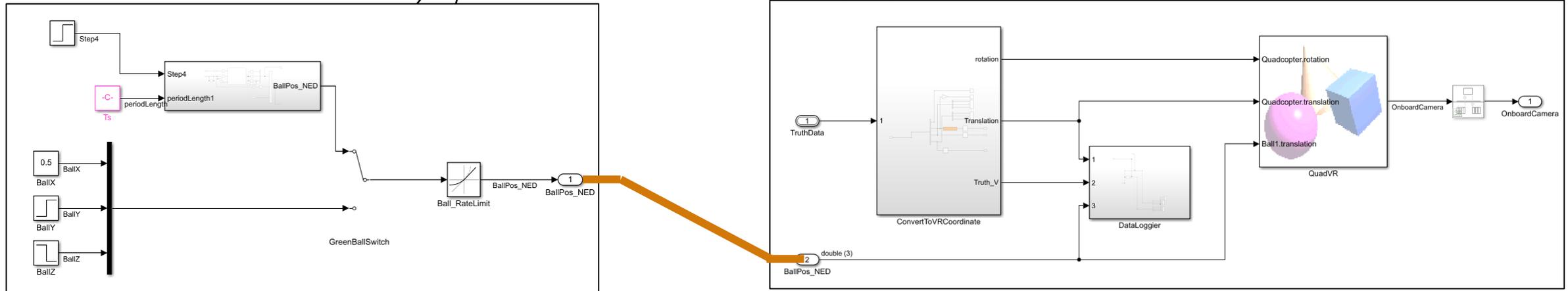
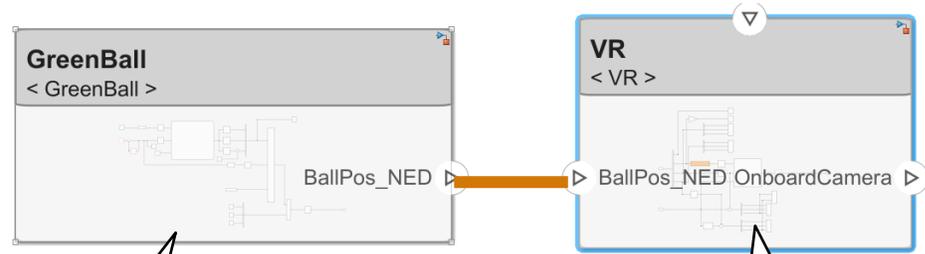


- コンポーネントモデルのI/Fレベルに到達すると、実装可能なレベルに要求が具体化されます
- そのレベルになればシステムモデル各々の下にSimulinkモデルを紐づけてモデリングを行うことが可能です

# Demo3: システム振る舞いの妥当性確認

モデル間で信号が接続されるため、システム全体をシミュレーション可能

コンポーネントモデル作成	システムレベルでの解析	インタフェース情報を継承した Simulinkモデルとの連携
システムを持つコンポーネントとコンポーネント間のインタフェースのモデリングが可能	コンポーネントモデルに定義した特性情報を用いた解析を MATLAB と連携して実施可能	アーキテクチャ設計で設計したインタフェース情報を継承したスクルトンモデルを生成可能



# Demo3: システム振る舞いの妥当性確認 (デモ動画)

## System ComposerモデルにSimulink参照モデルを紐づけ

**コンポーネントモデル作成**

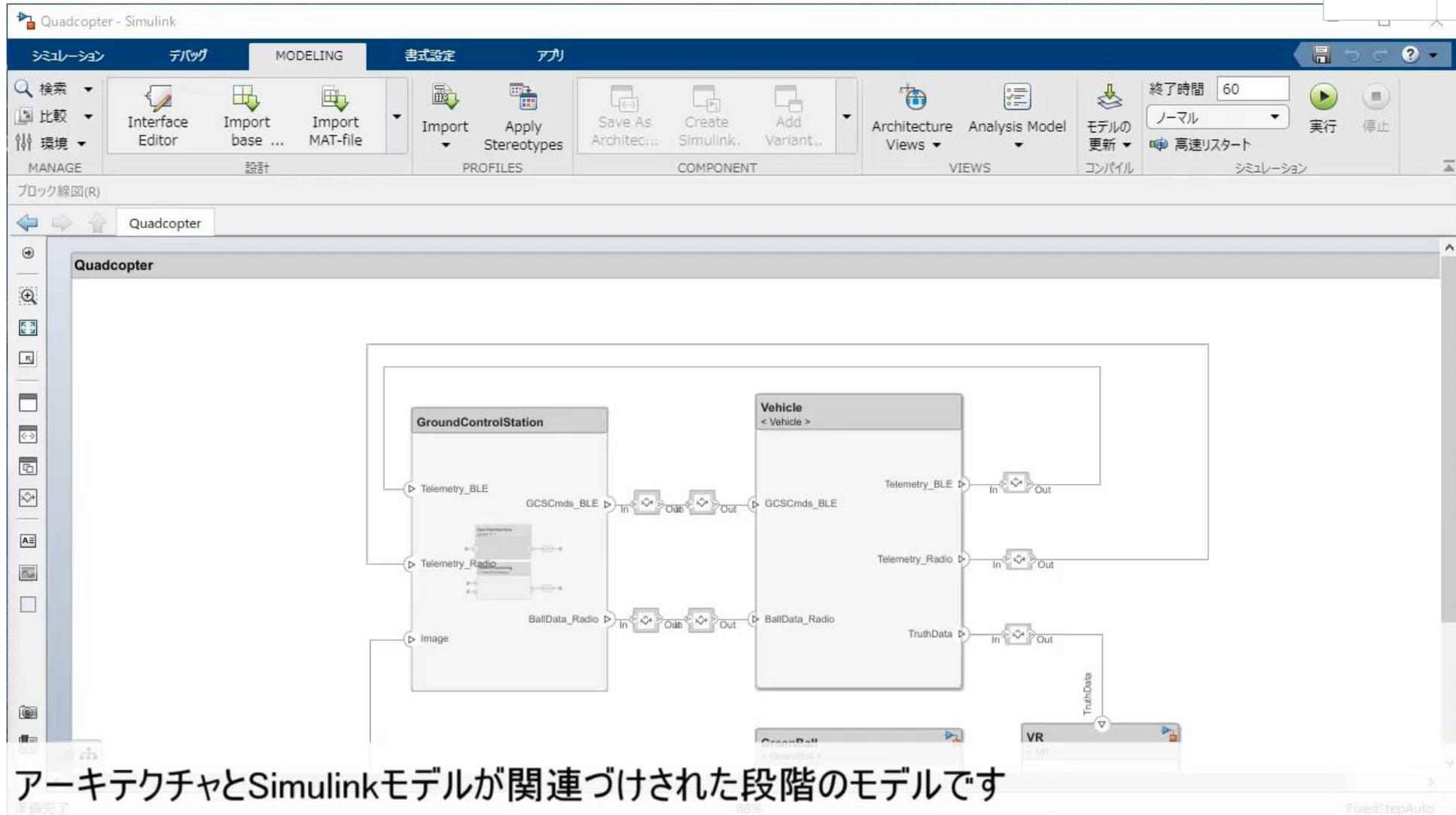
システム内コンポーネントとコンポーネント間のインタフェースのモデリングが可能

**システムレベルでの解析**

コンポーネントモデルに定義した特性情報を用いた解析をMATLABと連携して実施可能

**インタフェース情報を継承したSimulinkモデルとの連携**

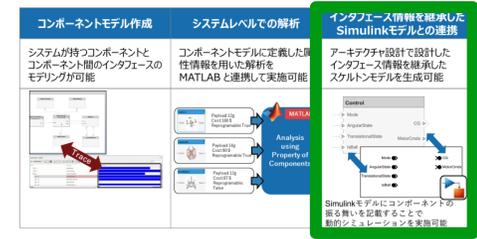
アーキテクチャ設計で設計したインタフェース情報を継承したスクリーンモデルを生成可能



アーキテクチャとSimulinkモデルが関連づけされた段階のモデルです

# Demo3: システム振る舞いの妥当性確認

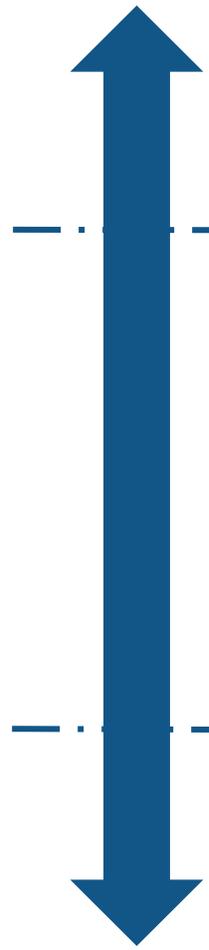
## 要求からモデルまでを一つの統合シミュレーション環境で表現可能



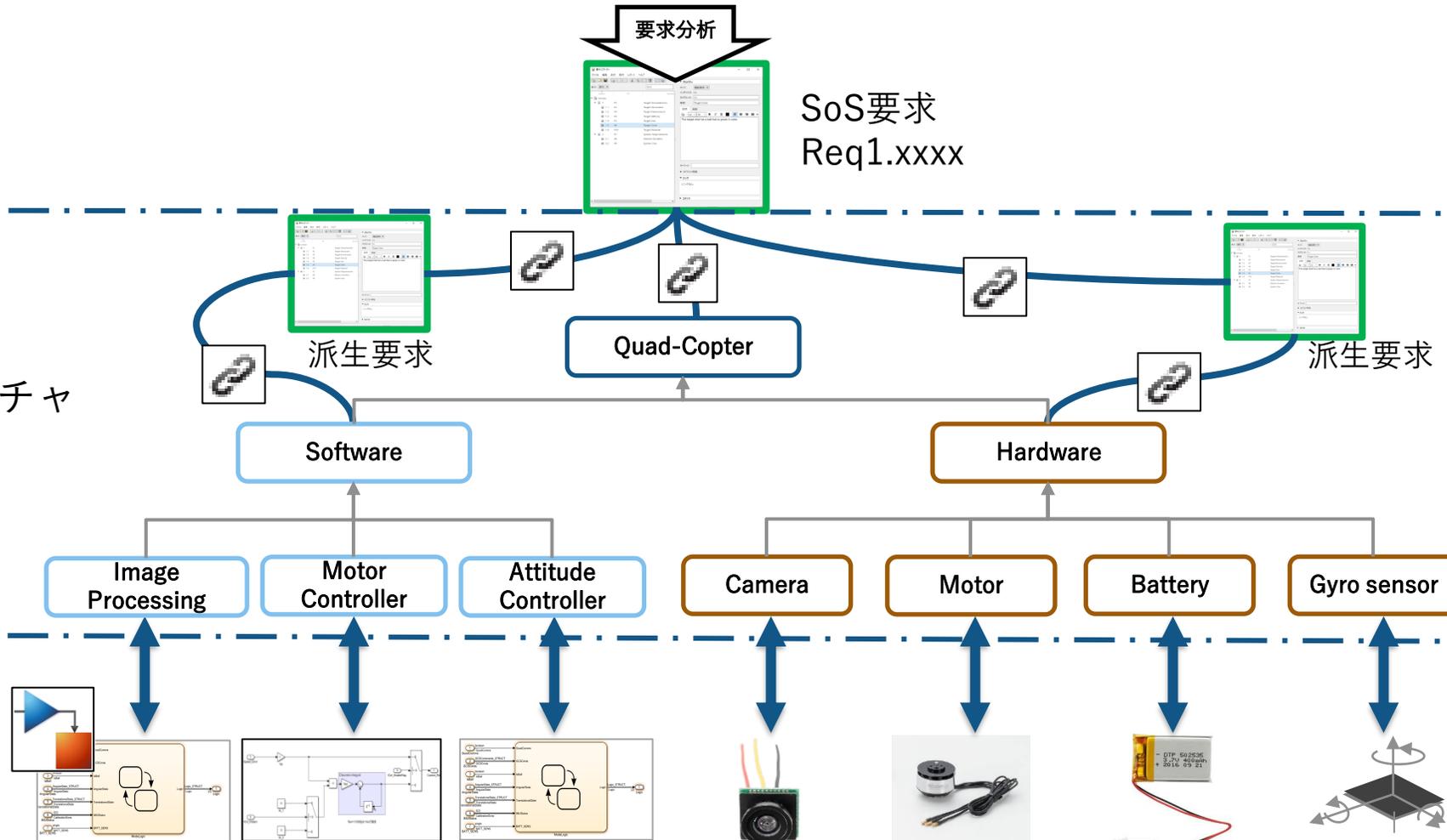
ステークホルダー要求:

緑のボールを追従するクアッドコプタ制御を可能な限り安価に構築すること

抽象度 高 : 要求



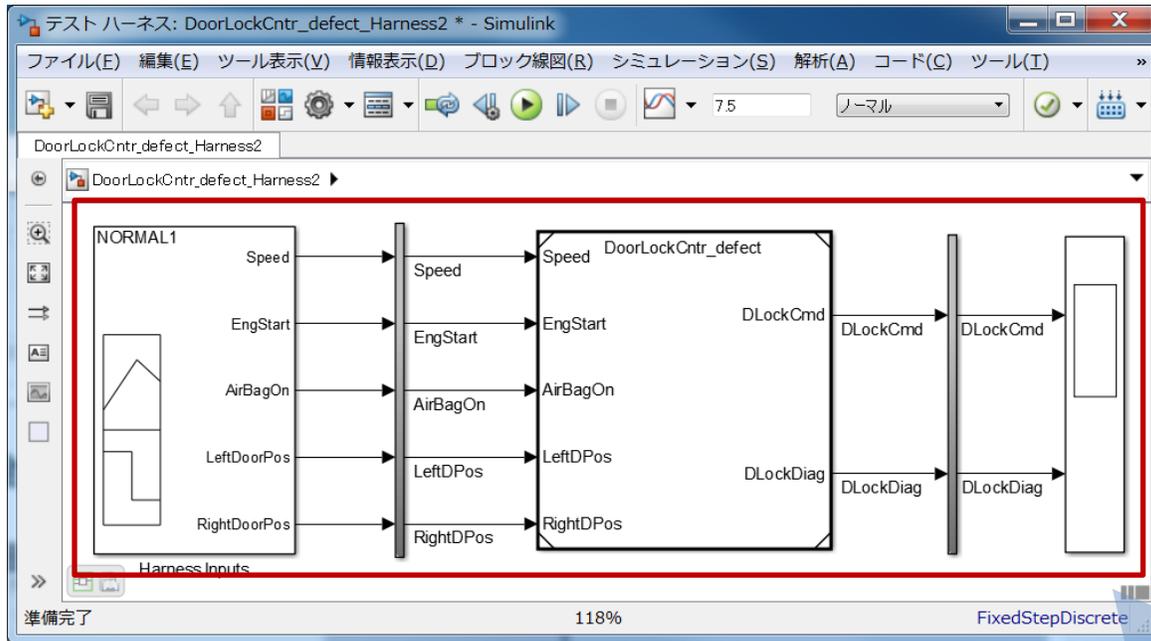
アーキテクチャ



抽象度 低 : Simulinkモデル

# Demo3: システム振る舞いの妥当性確認

## SimulinkTest テストマネージャーによるSoS要求検証



NAME	STATUS
Results : 2015-Nov-10 18:05:14	4
機能テスト:入力・出力期待値比較	4
Normal1	
Baseline Criteria Result	
DLockCmd	●
DLockDiag	○
AirBagOn	○
EngStart	○
LeftDPos	○
RightDPos	○
Speed	○
Sim Output (DoorLockCtr : r	
Normal2	
Fail1	

Simulink Test - テストマネージャー  
 テスト実行や合否判定と  
 レポート作成の自動化が可能

# Demo3: システム振る舞いの妥当性確認 (デモ動画)

## SimulinkTest テストマネージャーによるSoS要求検証



プロパティ	値
名前	1.1 Target Movement Test
タイプ	ベースライン テスト
モデル	Quadcopter
シミュレーション モード	[モデル設定]
場所	C:\Local\WS\Project\System...
有効	<input checked="" type="checkbox"/>
階層	QuadcopterSystemTests >

システム振る舞いの妥当性確認  
1. SimulinkTestのテストマネージャーに評価合格とする条件を定義

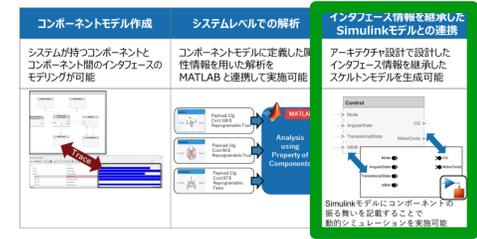
## まとめ

システムズエンジニアリングにおける課題をMathWorksツールが解決します

- ① アーキテクチャモデリングと、要求へのトレースが難しい/不可能
  - **Simulink Requirements™に要求を取り込みアーキテクチャ紐づけ可能**
- ② システムレベルでのアーキテクチャ検討方法が不明瞭
  - **ステレオタイプ情報を基にMATLABを用いて演算することが可能**
- ③ フィールドテストシミュレーション実行方法が不明瞭
  - **抽象度が異なるSoS要求からSimulinkモデルをまとめた統合環境を構築**
  - **SoS要求をシミュレーションによって定量的な値で動作検証が可能**

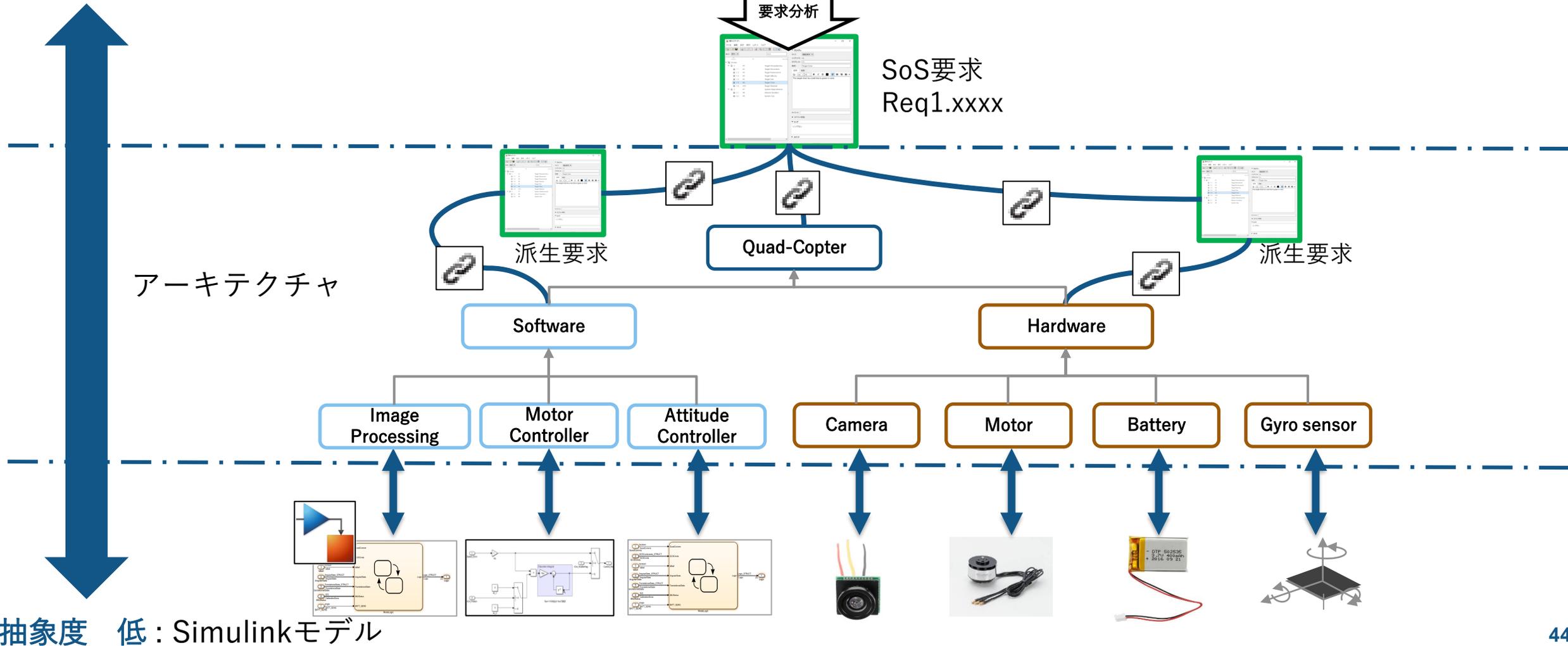
# Demo3: システム振る舞いの妥当性確認

## 要求からモデルまでを一つの統合シミュレーション環境で表現可能

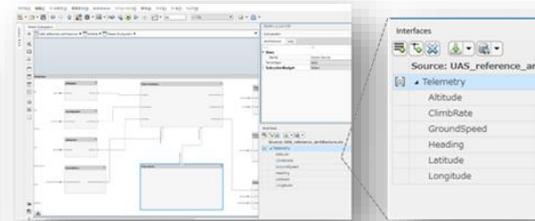
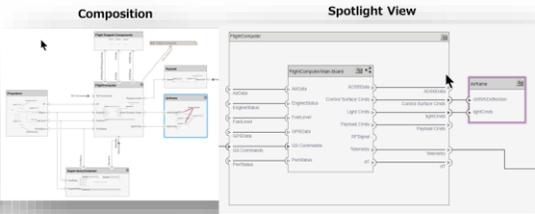
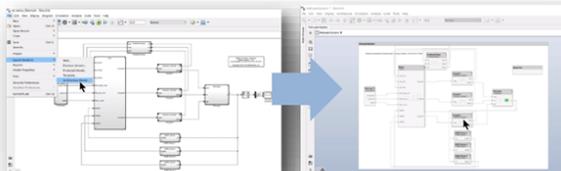
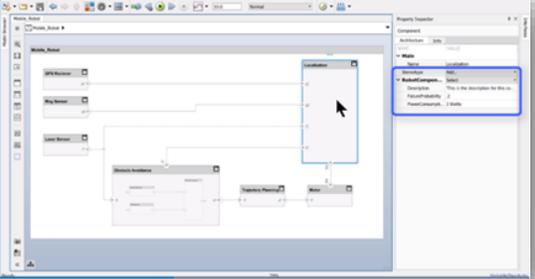
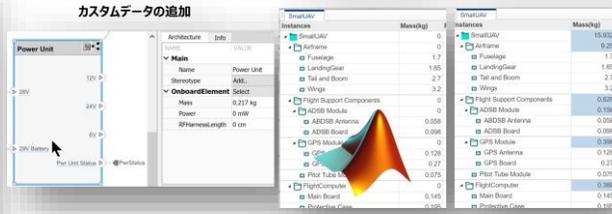


ステークホルダー要求:  
緑のボールを追従するクアッドコプタ制御を可能な限り安価に構築すること

抽象度 高 : 要求



# System Composer™はシステムおよびソフトウェア設計のためのアーキテクチャ設計と分析を可能にします

<h2>アーキテクチャオーサリング</h2>	<h2>インタフェース管理</h2>	<h2>Simulinkコンポーネントの作成とリンク</h2>	<h2>スポットライトビュー</h2>
<p>直感的な操作でコンポジションを作成できます</p>	<p>個別にインタフェースが定義できます</p>	<p>既存のSimulinkモデルとリンクが可能です</p>	<p>特定のコンポーネントを中心としたビュー機能を提供します</p>
 <p>Description == Architecture</p> <p>よりハイレベルな領域におけるシステム設計のスケッチが可能になります</p>	 <p>定義済みインタフェースとポートのリンクが可能です</p>	 <p>詳細設計領域のSimulinkモデルとシームレスに接続が可能です。</p>	 <p>マルチドメインシステムの設計をよりやり易くします。</p>
<h2>Simulinkモデルの再利用</h2>	<h2>ステレオタイプ</h2>	<h2>MATLABを利用した分析および最適化</h2>	<h2>アーキテクチャ参照</h2>
<p>既存のSimulinkモデルからアーキテクチャモデルの作成が可能です</p>	<p>ステレオタイプおよびプロパティの定義ができます</p>	<p>分析モデルの作成とMATLABによる分析が可能です</p>	<p>アーキテクチャモデルの参照化が可能です</p>
		 <p>データ分析にMATLABを用いることで、より実現性の高いアーキテクチャ設計が可能になります。</p>	

## 関連情報：

- [Simulink Requirement Webpage](#)
- [System Composer Webpage](#)
- [Online documentation & examples](#)
- [System Modeling and Simulation Webpage](#)
  
- [評価版の発行](#)



Accelerating the pace of engineering and science

© 2021 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See [www.mathworks.com/trademarks](http://www.mathworks.com/trademarks) for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

# Q&Aセッション

## 少々お待ちください

ご質問を確認する間、しばらく無音になりますが、  
ご遠慮なく**Q&A**パネルよりご質問をお送りください。

Q&Aパネルを閉  
じてしまった場合  
は、画面下部中央  
のメニューバーから  
再度開いてくだ  
さい。

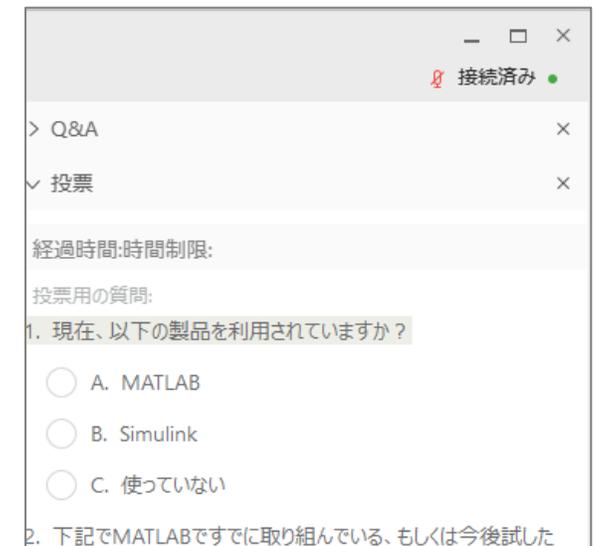


## 投票 (Polling) へご協力をお願い

投票 (Polling) パネルの簡単なアンケート  
にご協力ください。

所要時間1分程度です。

回答内容は他の参加者には共有されません



本日のセミナーの録画版は、1週間ほどで [jp.mathworks.com/videos](https://jp.mathworks.com/videos) に掲載されます。  
配布可能な資料がある場合は、該当ページの“コードとリソース”タブからダウンロードいただけます。