FMI 活用ガイド

Ver.1.0.1

2018 年 11 月 01 日

Revision	日付	主な内容
1.0.0	2018/10/01	Ver1.0.0 リリース
1.0.1	2018/11/01	6 章：6.2.4 を追加、6.2.5 の最少計算時間の誤記を修正

目次

本ガイド作成にあたり、自動車技術会 自動車制御とモデル部門委員会 FMI 活用・展開 WG に参加、協力頂いた個人、企業・団体は以下の通りです。

内田 裕美	株式会社 ISID エンジニアリング
市原 純一	AZAPA 株式会社
関末 崇行	アンシス・ジャパン 株式会社
猿木 恭文	エムエスシーソフトウェア 株式会社
村上 晋太郎	慶応義塾大学大学院
岩ヶ谷 崇	サイバネットシステム 株式会社
緒方 洋介	シーメンス 株式会社
恒光 千恵	ダッソー・システムズ 株式会社
VIRY, Guillaume	ダッソー・システムズ 株式会社
小池 理	dSPACE Japan 株式会社
三輪 修也	株式会社 デンソー
平野 豊	トヨタ自動車 株式会社
守屋 一成	株式会社 豊田中央研究所
斉藤 春樹	日産自動車 株式会社
広野 友英	ニュートンワークス 株式会社
神野 英章	株式会社 本田技術研究所
平尾 俊一	株式会社 本田技術研究所
赤阪 大介	The MathWorks GK
小森 賢	マツダ 株式会社
寺岡 陽一	マツダ 株式会社
佐藤 裕司	三菱スペース・ソフトウエア 株式会社
GAO, Rui	モデロン 株式会社
西田 怜美	モデロン 株式会社

（企業・団体名で 50 音順）

第 1 章　本ガイドについて

1.1　本ガイドの目的

本ガイドは自動車システムの開発においてシミュレーションを活用しているエンジニアを対象としたシミュレーションモデルの交換および接続のための手引き書です。

自動車システムは高機能と高性能が求められ、その複雑なシステムの開発においてシミュレーションを活用したモデルベース開発の重要性が増しています。そして、複数のシステムや部品を組み合わせてシステム全体の振る舞いを検証するために、社内の部署間あるいは他社とシミュレーションモデルの流通が求められています。

本ガイドではシミュレーションモデルを交換および接続するための基礎的な技術や注意点を解説しています。

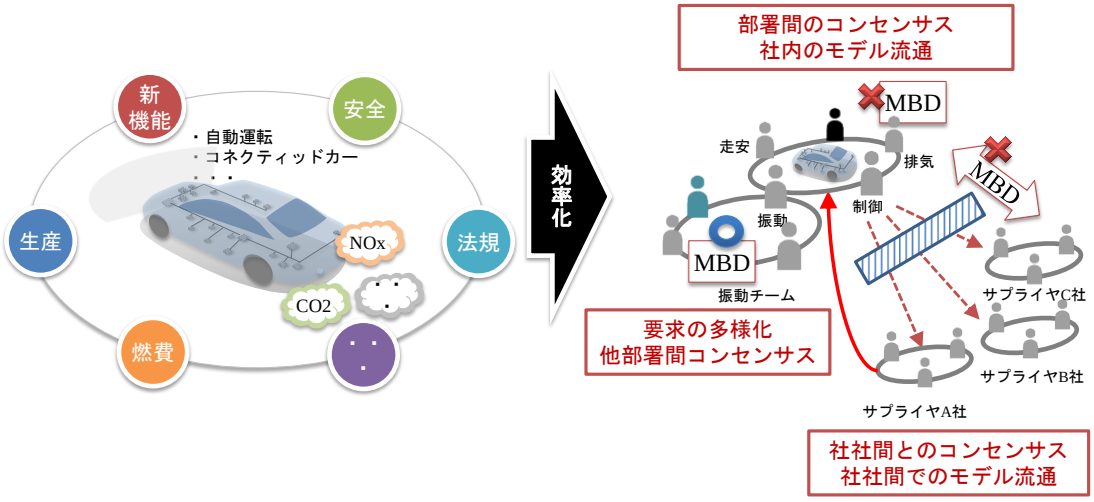


図 1-1　モデル流通のイメージ

（自技会　振動騒音シンポジウム（2017）、「FMI によるモデル流通を目指して」より引用）[1]

1.2　はじめに

自動車システムの開発においてシミュレーションは設計／試作／テストの一連の全ての工程で活用されています。コンピュータの性能向上とソフトウェアの進歩に伴い、シミュレーションの活用が進んでいます。モデルを用いたシミュレーション環境の構成要素は下記です。

表 1-1　シミュレーション環境の構成要素

モデル	シミュレーション対象の物理動作やコントローラ動作を 下記により抽象化したもの ・グラフィカルなモデル記述 ・モデリング言語やプログラミング言語 ・数式記述、表、マップなど
ソルバ	積分等の数値演算を行うための機能
アプリケーションソフトウェア	シミュレーションツールの動作プログラム
オペレーティングシステム	コンピュータの基本プログラム
コンピュータ	プログラムを実行する物理装置、ネットワーク通信環境も含む

1.3　背景

1.3.1　モデル流通への期待

自動車システムの開発は、まずシステムの構成要素となる部品の設計／試作を行い、次にそれらを組み合わせた試作車を用いてテストが行われてきました。しかし、高機能化と高性能化による複雑化と開発期間短縮への要望に応えられなくなり、従来型の開発スタイルでは限界が見えてきました。そこで、仕様書と実部品という関係から、仕様書とそれに対応するシミュレーションモデルへの置き換えが進み、システムの構成要素となるサブモデルを組み合わせた試作車モデルを用いたシミュレーションでテストを行うモデルベース開発へ変わりつつあります。そして、社内の部署間あるいは他社とのシミュレーションモデルの流通が求められています。

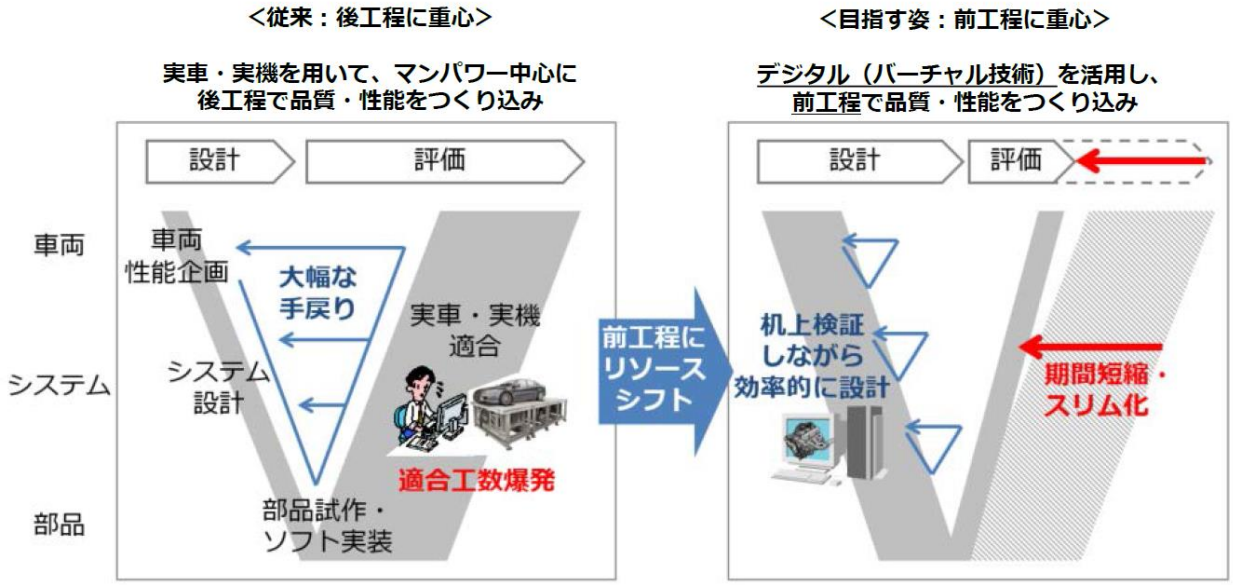


図 1-2　「車」の作り方の革新

（経済産業省　自動車新時代戦略会議（第 1 回）配布資料より引用）[2]

1.3.2　モデル流通の課題

シミュレーションモデルの記述には、様々なシミュレーションツールが使用されているため、企業内の部門間あるいは企業間でモデルを流通するには、相互のシミュレーションツールの統一か、異なるシミュレーションツールにより作成されたモデルの接続が必要です。特に、自動車システム開発に関係する全ての企業でシミュレーションツールを統一するのは現実的でないため、異なるシミュレーションツール間でのモデルの交換および接続するためのインターフェースの共通化が課題となっています。

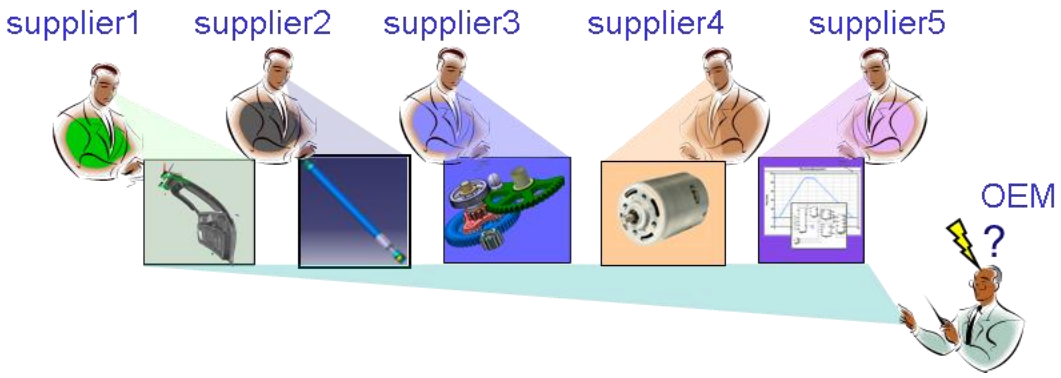


図 1-3　OEM とサプライヤ間モデル流通

（8th International Modelica Conference 2011 前刷り資料より引用）[3]

1.3.3　モデル流通の標準化

シミュレーションツールに依存しないモデル接続のための共通インターフェースとして、欧州の公的プロジェクトが規格化した FMI（Functional Mock-up Interface）が世界的に普及しています。自動車技術会でも 2012 年 3 月に「国際標準記述によるモデル開発・流通検討委員会」を設置し、モデル接続技術検討 WG において FMI によるモデル接続検証と、モデルの交換および接続の推進が行われてきました。そして、その活動を踏まえ 2015 年 3 月に「非因果モデリングツールを用いた FMI モデル接続ガイドライン Ver.1.0 」[4] を発行しました。

その後、この活動は「自動車制御とモデル部門委員会」の FMI 活用・展開検討 WG に引き継がれ、非因果モデリングツールに依存せずに異なるシミュレーションツール間でモデルの交換および接続する一般的な手法を検討しています。

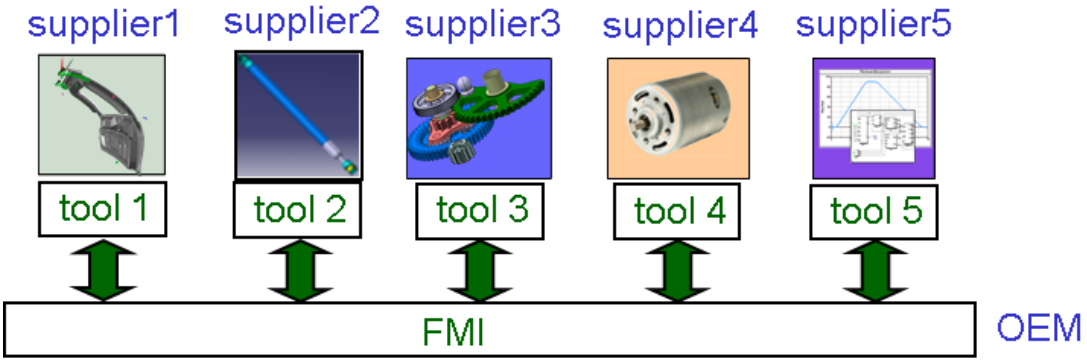


図 1-4　FMI による OEM とサプライヤ間モデル流通
（8th International Modelica Conference 2011 前刷り資料より引用）[3]

第 2 章　FMI の基本

この章では、FMI に関する基本的な知識や情報について説明します。

2.1 は、FMI の成り立ちについて説明します。2.2 では、FMI の 2 つの動作モードである Model Exchange と Co-Simulation の基本構造について、2.3 及び 2.4 では、それぞれの基本構成と特徴について解説します。2.5 では FMI をサポートするシミュレーションツールの確認方法を掲載しています。

2.1　FMI の成り立ち

FMI とは、ツール間連携のための標準インターフェースです。FMI は、2010 年に DAG(Daimler AG)を中心とした ITEA2 の MODELISAR プロジェクト（約 30M€のプロジェクト予算）で、様々なツール間を接続するために策定された標準インターフェース仕様です。

FMU（Functional Mock-up Unit）という単位を一つのモデルとし、ツール間で交換したり、接続したりすることが可能です。その後、2011 年から Modelica Association （モデリカ協会）のプロジェクトの一つとなり、改定と保守が継続的に行われています。

FMI の規格は、v1.0 において Model Exchange と Co-Simulation は別々に企画・策定されており 2.2 で後述する model Description File の構造は共通ではありません。2011 年に MA Project（モデリカ協会プロジェクト）になり、共通の構造をもつ v2.0 が企画され、策定されてきました。

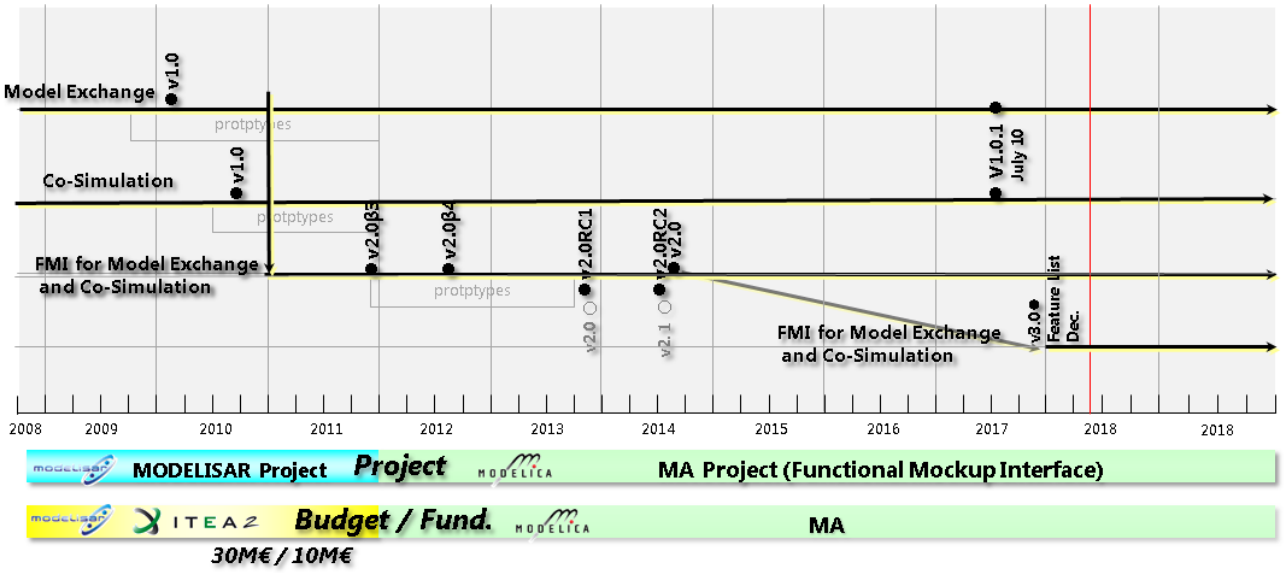


図 2-1　FMI プロジェクトの歴史と規格

（モデリカ協会情報を元に新規作成）

2.2　FMU 動作モードと構造

2.2.1　FMU の動作モード

FMU の動作モードは、Model Exchange (ME) と Co-Simulation (CS)の 2 つの動作モードがあります。動作モードの大きな違いは、ソルバが FMU を取り込むマスタツールにあるか、FMU に内包するかになります。

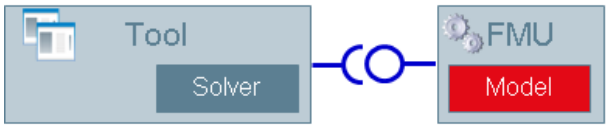


図 2-2a　FMU にソルバを持たない ME

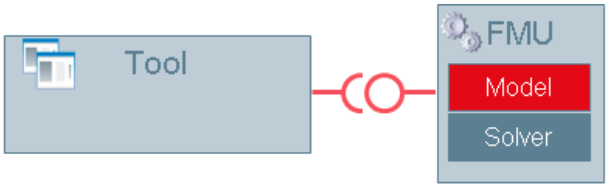


図 2-2b　FMU にソルバを内包する CS

(8th International Modelica Conference 2011 講演資料より引用) [5]

2.2.2　FMU のファイル構造

FMU の拡張子は.fmu であり、その実態は ZIP 形式で圧縮されたものです。概念を図 2-3 に示します。FMU は、Windows OS における実行形式の dll や Linux OS における実行形式の so とモデルの詳細説明である xml 形式の model Description File からなりたっています（5.3.2 参照)。

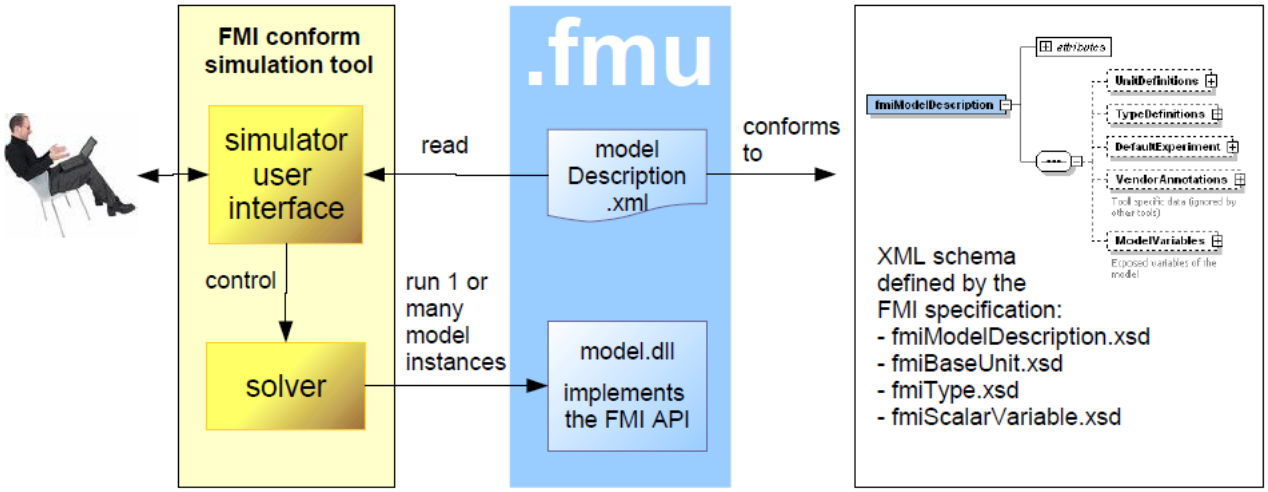


図 2-3　FMU の構造

(8th International Modelica Conference 2011 前刷り資料より引用) [3]

2.2.2.1　model Description File の構造

model Description File の構造を図 2-4 に示します。先に述べたように、FMI1.0 では ME と CS は企画段階から別々に立案・策定され、共通の構造がないので、FMI2.0 の構造を用いて説明します。

なお、FMI 1.0 と 2.0 との間には、互換性は保たれていないので取り扱いには注意が必要です。

はじめに、ME か CS かの区別の記述があり、その後に実装、単位、変数、属性、構造とその属性などが細かく記述されています。

実行形式の dll を生成するためにはソースファイルが必要です。FMU を生成するツールでは、dll 生成後にソースファイルを削除し、隠蔽可能としています。この設定方法は、ツールに依存しますので、使用するツールのマニュアルを参考にして下さい。

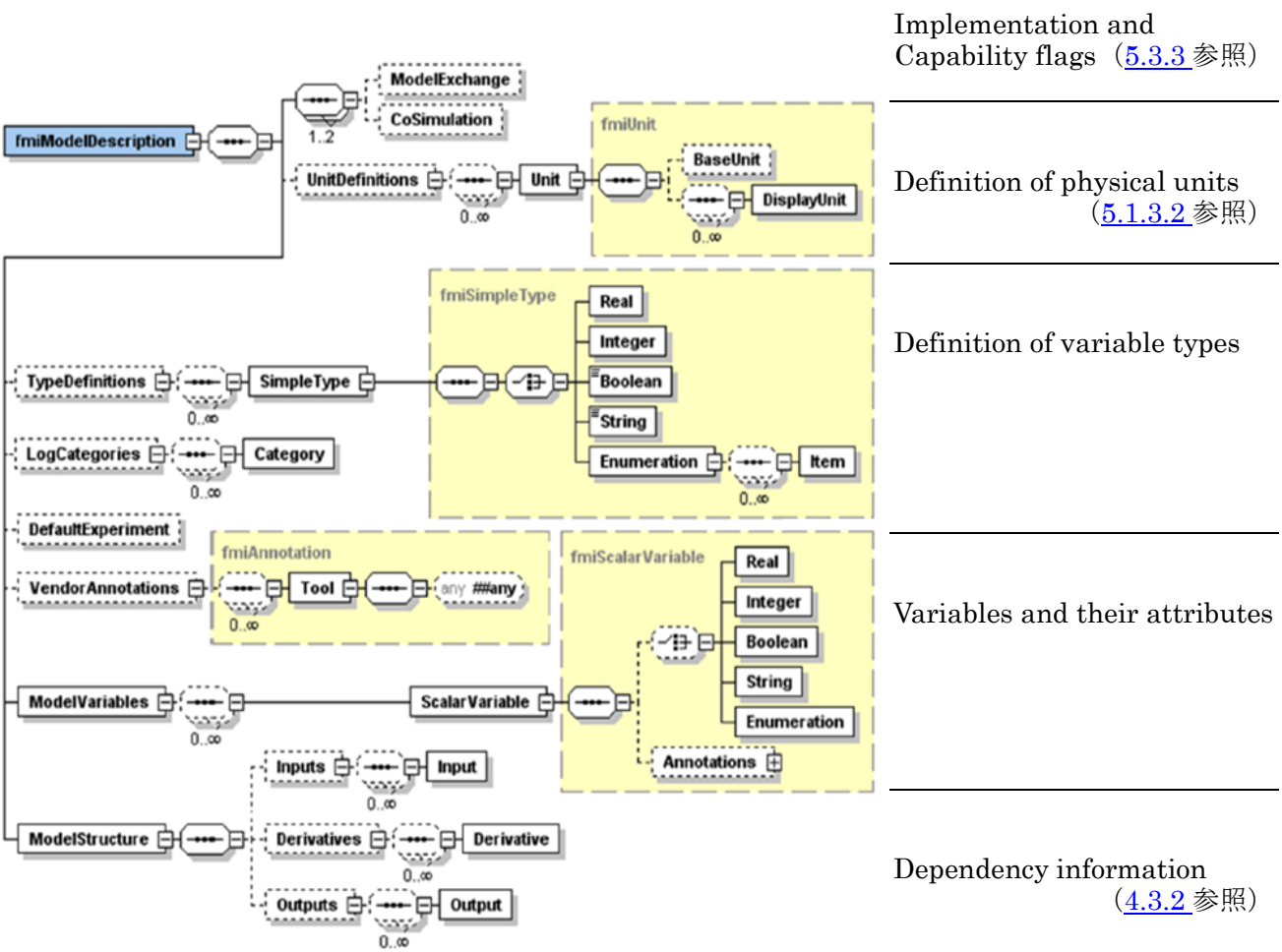


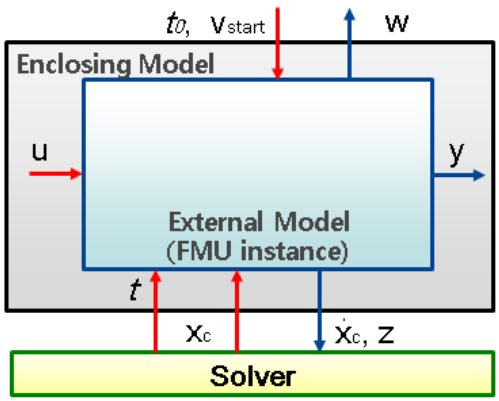
図 2-4 model Description File の構造

（2nd Japanese Modelica Conference 基調講演資料より引用）[6]

2.3　Model Exchange（ME）の構成と特徴

2.3.1　ME における信号の流れ

ME モードでは FMU を取り込んだシミュレーションツールのソルバを用いて FMU 内の計算は行われます。従って、FMU とその周辺のモデルの時間遷移が同一のソルバで同時に実施されるため、時間同期が保証されます。



t	time
$\mathbf{v}_{start}$	variable
$\mathbf{u}(t)$	inputs (continuous-time and/or discrete-time)
$\mathbf{y}(t)$	outputs (continuous-time and/or discrete-time)
$\mathbf{w}(t)$	local variables (continuous-time and/or discrete-time)
$\mathbf{z}(t)$	event indicators (continuous-time)
	Elements of local variables w and/or output y:
$\mathbf{x}_c(t)$	continuous-time states (continuous between events)

図 2-5　ME モードの信号の流れ

（FMI 仕様書：FMI for ModelExchange and CoSimulation v2.0 より引用）[7]

青矢印（→）：FMUからの出力情報　　赤矢印（→）：FMUへの入力情報

Vstart , u , y , w , xd は Real, Integer, Boolean, String 型;　　t , xc , z は Real 型

2.3.2　ME での適用事例

図 2-6a に非因果系ツールに取り込み接続した FMI の例を示します。赤色破線がそれぞれ FMU になっています。図 2-6b に構造の概念図を示します。3 つの FMU を取り込んだツール上のソルバでシミュレーションが実施されるのが理解できると思います。

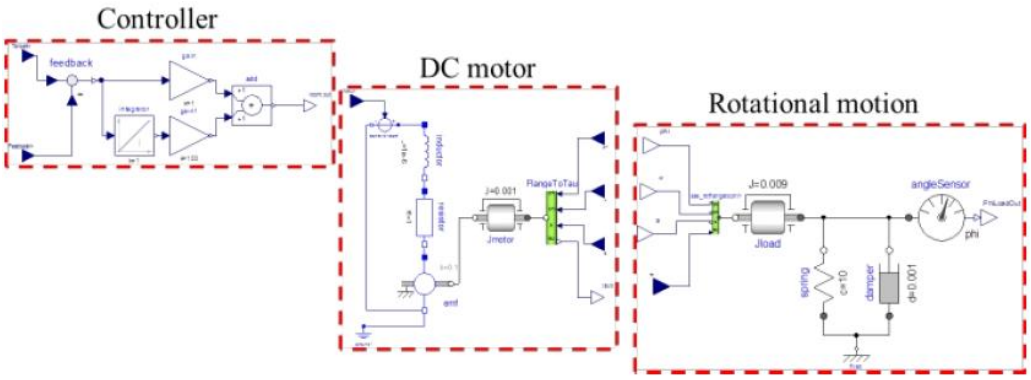


図 2-6a　Model Exchange モードでの適用事例

（2nd Japanese Modelica Conference 講演資料より引用）[8]

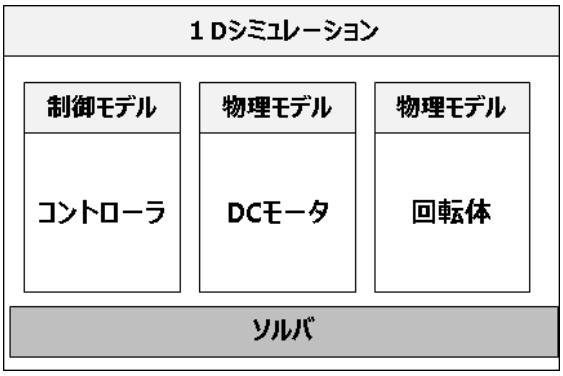


図 2-6b　ME モデル（2-6a）の補足

2.4　Co-Simulation（CS）の構成と特徴

2.4.1　CS の 3 つの形態

CS は、運用により 3 つの形態に分類されます。

もっとも基本的なものが Stand Alone（図 2-7）であり、PC 上でマスタとなるツールと FMU が 1 つのコアの上で動作します。

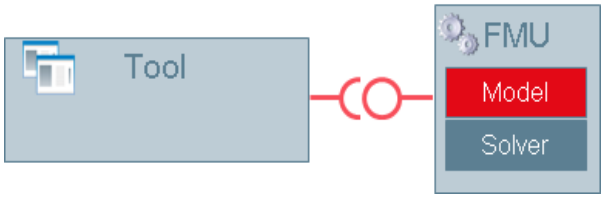


図 2-7　Stand Alone Co-Simulation

（8th International Modelica Conference 2011 講演資料より引用）[5]

2 つ目の形態は、Tool Coupling（図 2-8）であり、マスタとなるツールと FMU とが別々のプロセッサもしくは別々のコアの上で動作します。

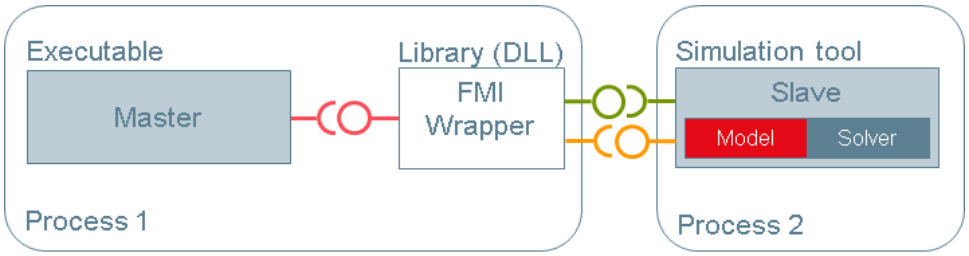


図 2-8　Tool Coupling Co-Simulation

（8th International Modelica Conference 2011 講演資料より引用）[5]

3 つ目の形態は、Distributed（図 2-9）で、莫大なコンピュータリソースを必要とするような大規模システムで適用されます。クライアント・サーバ方式の基本的な分散環境の上に、FMI 規格に準拠した構成になっています。

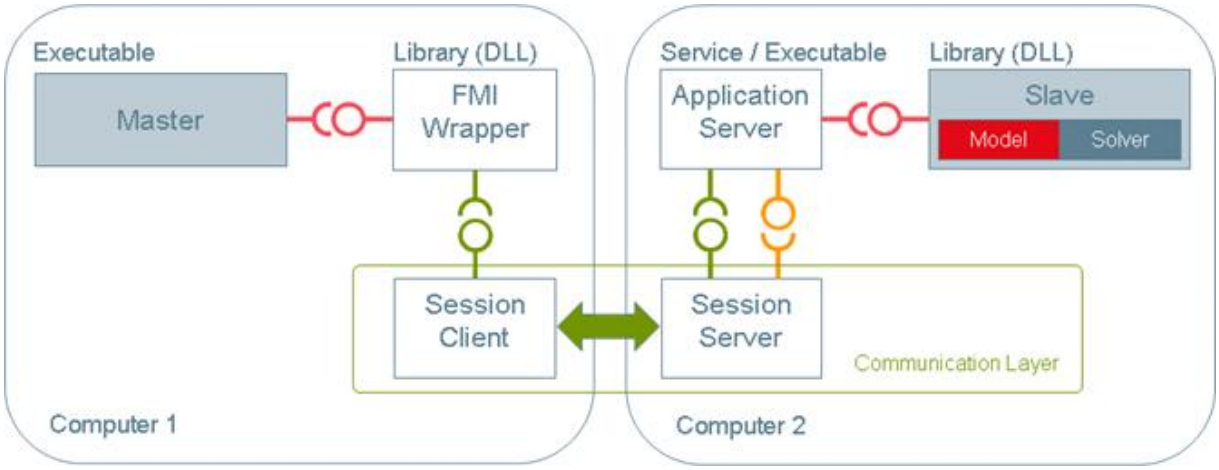
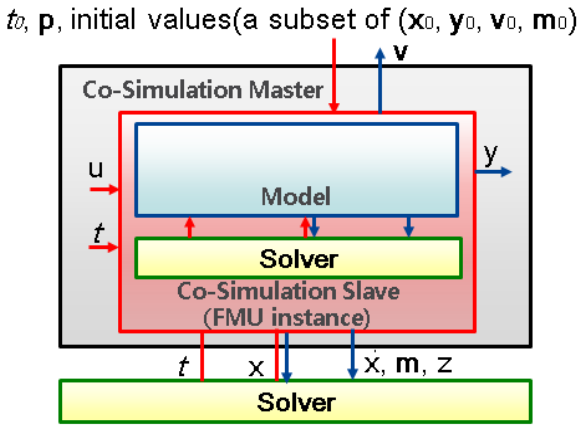


図 2-9　Distributed Co-Simulation

（8th International Modelica Conference 2011 講演資料より引用）[5]

2.4.2　CS における信号の流れ

CS における信号の流れを図 2-10 に示します。(u)、(y)が FMU の入出力であり、FMU 内部のソルバによって計算されます。この入力(u)や出力(y)は、第 3 章にて後述する Communication Step Size で設定された時間ごとに FMU を取り込んだツールとの間で信号授受を行います。



t	time
m	discrete status(constant between events)
p	parameters of type Real , Integer, Boolean, String
u	inputs of type Real, Integer, Boolean, String
v	all expose variables
x	continuous status (continuous between events)
y	outputs of type Real, Integer, Boolean, String
z	event indicators

図 2-10　CS モードの信号流れ

（FMI 仕様書：FMI for ModelExchange and CoSimulation v2.0 より引用）[7]

Co-Simulation では、以下 2 つの基本的な機能を実現する必要があります。

1. サブシステム間のデータ受け渡しの機能
2. 全てのサブシステムのシミュレーションを同期し、Communication Step を続行する機能

$tc_i \rightarrow tc_{i+1}$ from initial time $tc_0 := t_{start}$ to end time $tc_N := t_{stop}$.

2.4.3　CS での適用事例

図 2-11a に Co-Simulation で接続された FMU と各 FMU 間での信号の流れを示します。ここでは各信号が直接 FMU 間で授受されているように見えますが、実際は図 2-11b に示すように、取り込んだツールを介して信号授受を行います。各 FMU は独自のソルバを持ち、それぞれ計算を行い、Communication Step Size 毎に取り込みツールとこのソルバを介して信号授受することにより、全体のシミュレーションを実行します。

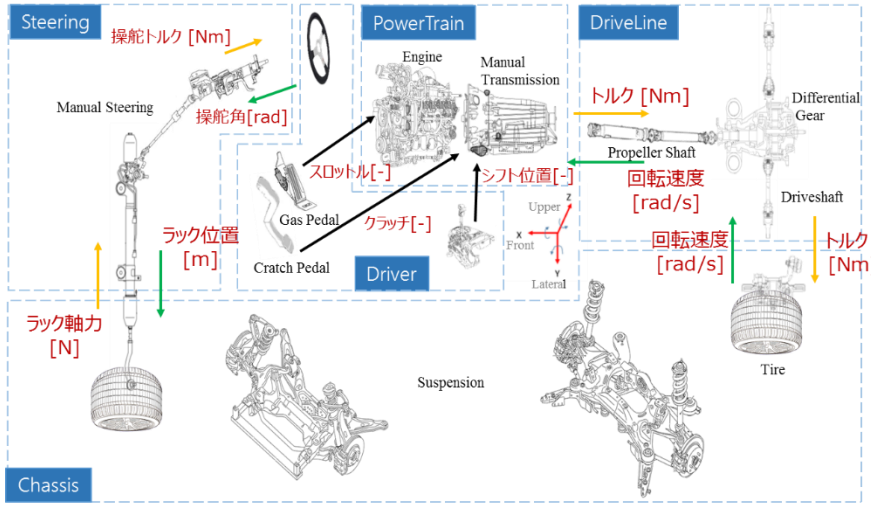


図 2-11a　Co-Simulation モードの適用事例

（2nd Japanese Modelica Conference 講演資料より引用）[8]

具体的には図 2-11a のドライバとステアリングに対する信号の送受信は、図 2-11b に示すように、ドライバが操舵角を出力し、FMU の取り込みツールを介してステアリングに送信しされます。操舵角に応じた操舵角トルクがステアリングの FMU 内で計算され、この結果が取り込みツールを介してドライバに渡されます。この受け渡しのタイミングは、Communication Step Size（CSS）で設定された時間間隔になります。ドライバからの操舵角出力は、取り込みツールを介していますので、ステアリングが取り込みツールから受け取れるのは次の CSS のタイミングになります。これを時間軸で表現すると図 2-11c のようになります。なお、この適用事例はチュートリアルを用意していますのでご活用下さい（6.2 参照）。

図 2-11b　CS モデル（2-11a）の補足

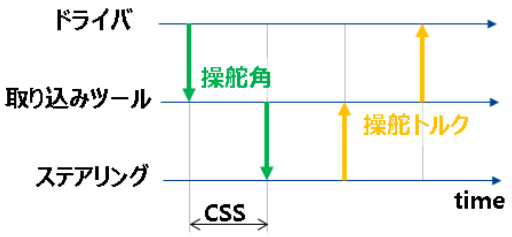


図 2-11c　CS モデル（2-11a, b）の補足

FMI は、ME や CS を単独で使用するだけでなく、組み合わせて使用することも可能です。ツールも 3D ツールや 1D のツールに依存せず混在して使用可能です。

図 2-12 は、DAG（Daimler AG）が 2 つの ME の制御モデル FMU を 1D シミュレーションツールの上に取り込み、さらに 3D 構造解析ツールに取り込んで使用したものとなります。

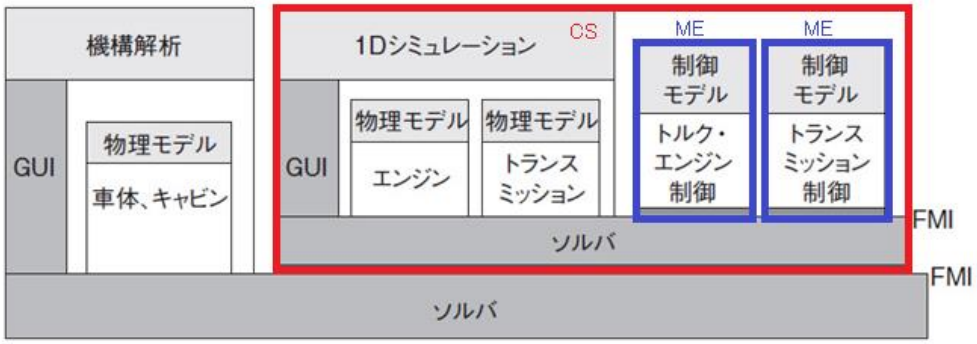


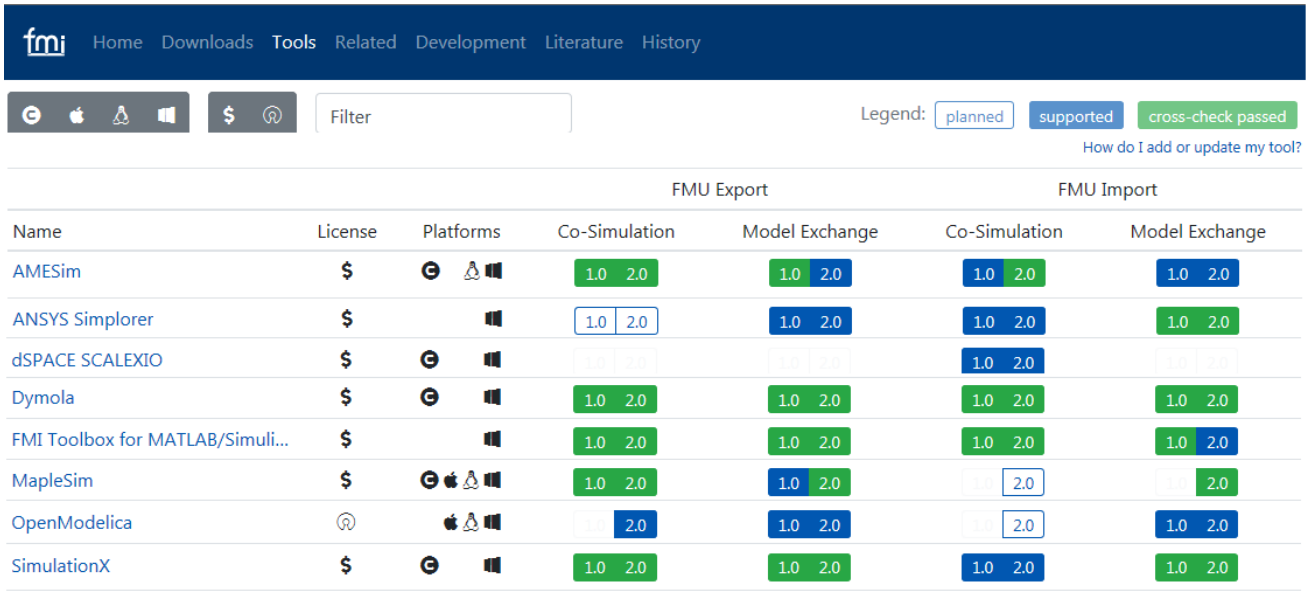
図 2-12 独 DAG 社における FMI 応用プロジェクト
（日経デジタルヘルス HP より引用、一部加筆）[9]

ここでは、商用車のギヤのシミュレーションを、複数のツールを用いて実行しています。図 2-12 の右から、トランスミッションの制御モデル（独自プログラム）とエンジン制御モデル（「Simulink」上に作成）を 1D シミュレーションツールの物理モデルに取り込みます。さらに、これを機構解析ツールの車体の物理モデルに取り込み評価を実施した例です。

2.5　ツールのサポート状況

FMI をサポートする代表的なツールを表 2-1 に示します。

表 2-1　FMI をサポートするツール（FMI ホームページより抜粋）



Name	License	Platforms	FMU Export: Co-Simulation	FMU Export: Model Exchange	FMU Import: Co-Simulation	FMU Import: Model Exchange
AMESim	$		1.0 2.0	1.0 2.0	1.0 2.0	1.0 2.0
ANSYS Simplorer	$		1.0 2.0	1.0 2.0	1.0 2.0	1.0 2.0
dSPACE SCALEXIO	$				1.0 2.0	
Dymola	$		1.0 2.0	1.0 2.0	1.0 2.0	1.0 2.0
FMI Toolbox for MATLAB/Simuli...	$		1.0 2.0	1.0 2.0	1.0 2.0	1.0 2.0
MapleSim	$		1.0 2.0	1.0 2.0	2.0	2.0
OpenModelica			2.0	1.0 2.0	2.0	1.0 2.0
SimulationX	$		1.0 2.0	1.0 2.0	1.0 2.0	1.0 2.0

ホームページ上には、以下の対応状況が掲載されています。

Legend: [planned] 対応計画中、[supported] 対応済み、[cross-check passed] クロスチェック済み

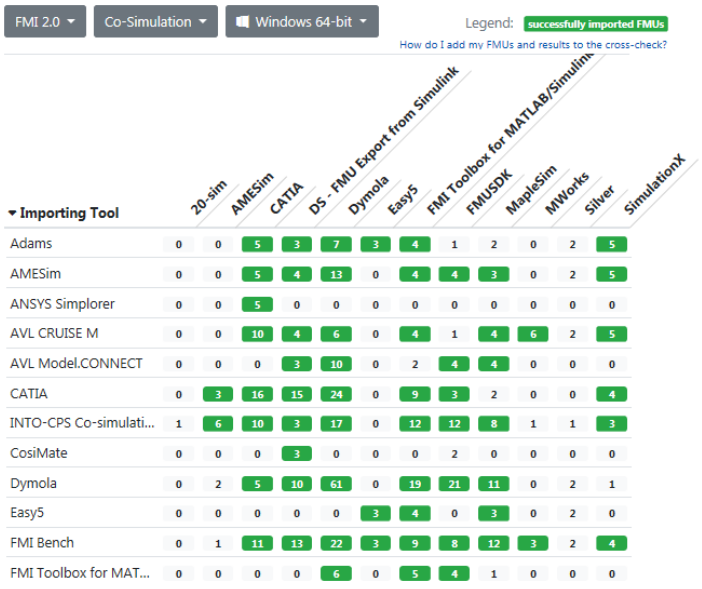
Licence：ライセンスの形態（商用、オープンソース）

Platforms：対応するプラットフォーム（C コード、macOS、Linux、Windows）

FMU Export／Import：FMU の生成、取り込み（CS、ME それぞれの FMI1.0／2.0 への対応）

表 2-2 は、FMI プロジェクトで定義された評価モデルを用いて、どのツールとの組み合わせで正常に動作したか結果が掲載されています。3 つ以上クロスチェックが確認できると表示が[緑色]になります。

表 2-2　クロスチェックの結果：FMI2.0 Co-Simulation Windows64bit（FMI ホームページより抜粋）



Importing Tool	20-sim	AMESim	CATIA	DS - FMU Export from Simulink	Dymola	Easy5	FMI Toolbox for MATLAB/Simulink	FMUSDK	MapleSim	MWorks	Silver	SimulationX
Adams	0	0	5	3	7	3	4	1	2	0	2	5
AMESim	0	0	5	4	13	0	4	4	3	0	2	5
ANSYS Simplorer	0	0	5	0	0	0	0	0	0	0	0	0
AVL CRUISE M	0	0	10	4	6	0	4	1	4	6	2	5
AVL Model.CONNECT	0	0	0	3	10	0	2	4	4	0	0	0
CATIA	0	3	16	15	24	0	9	3	2	0	0	4
INTO-CPS Co-simulati...	1	6	10	3	17	0	12	12	8	1	1	3
CosiMate	0	0	0	3	0	0	0	2	0	0	0	0
Dymola	0	2	5	10	61	0	19	21	11	0	2	1
Easy5	0	0	0	0	0	3	4	0	3	0	2	0
FMI Bench	0	1	11	13	22	3	9	8	12	3	2	4
FMI Toolbox for MAT...	0	0	0	0	6	0	5	4	1	0	0	0

ホームページは日々更新されていますのでご確認下さい。

第 3 章　使ってみよう、作ってみよう

この章では、実際に FMI に触れてみましょう。3.1 では、簡単なサンプル FMU モデルを用いて取り込みからシミュレーション実行までの流れを体験してください。3.2 では、FMU を生成する際の基本的な注意事項について説明しています。

3.1　使ってみよう

3.1.1　サンプルモデルを準備しよう

自動車技術会ホームページの自動車制御とモデル部門委員会ページに用意したサンプルモデルを実際に取り込んでシミュレーションを実行してみましょう。モデルは、2018 年 5 月の自動車技術会春季大会にて、当 WG で講演した際のベンチマークモデルを使用します。[12]

ファイル名：Sample_3p1.zip　：ZIP ファイルには以下 4 個のファイルが格納されています。

FMU1 (Model Exchange)	: FMU_J1_Case1_Dymola_ME_2.fmu
FMU2 (Model Exchange)	: FMU_SD1_Case1_Dymola_ME_2.fmu
FMU1 (Co-Simulation)	: FMU_J1_Case1_Dymola_CS_2.fmu
FMU2 (Co-Simulation)	: FMU_SD1_Case1_Dymola_CS_2.fmu

今回の例題は，簡単な回転ばねダンパモデルを分割したもので、FMU は下記にて作成されています。

Version	: FMI 2.0
Type	: ME および CS
Compiler	: Visual C++ 64bit
OS	: Windows 64bit

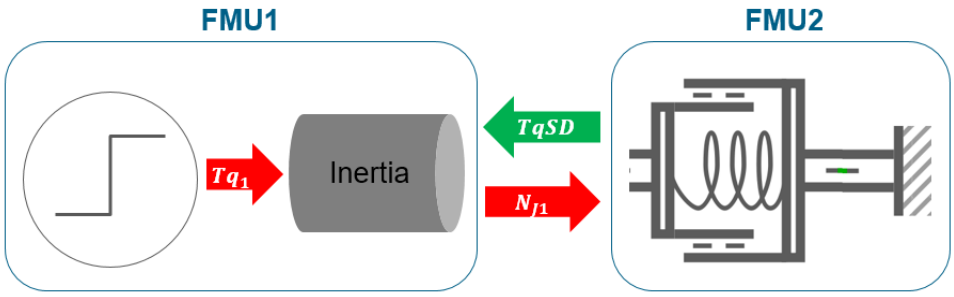


図 3-1　モデルの構成

まずは、Import に対応したシミュレーションツールを用意する必要があります。2.5 で紹介した FMI ホームページにてお手持ちのツールの FMU Import 対応状況を確認してください。

ツールが用意できれば、まずは FMU を取り込みますが、その手順は使用ツールよって異なりますので、ここでは割愛します。

うまく取り込めたでしょうか？

以降は、Simcenter Amesim での実行例をご紹介いたします。

3.1.2　Model Exchange でシミュレーションしよう

3.1.2.1　標準パラメータで実行

次に、取り込んだ FMU を接続していきます。まずは、FMU1、FMU2 ともに Model Exchange を使います。それぞれの FMU には 1 入力、1 出力のみである為、双方の FMU の出力と入力をつなげば、問題なく図 3-2 のように接続できるはずです。

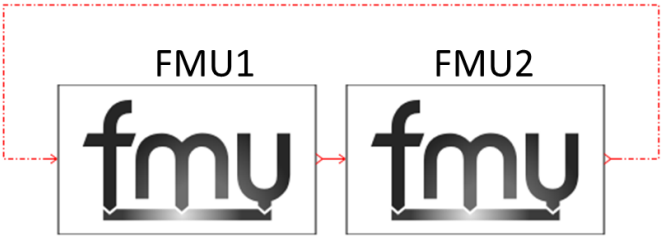


図 3-2　FMU の接続

次に FMU のパラメータを設定します。

図 3-3a、3-3b のパラメータ”path to the unzipped FMU root”（ツールにより異なります）は、解凍した FMU の dll 格納場所を指定します。シミュレーション実行時にはこのアドレスにある dll を呼ぶのでパスが重要です。

#から始まる値は積分計算を行う状態変数の初期値です。変更可能ですが、まずはこのままの設定とします。#などの変数は使用するツールにより表示が異なるのでご注意ください。

Parameters

Title	Value	Unit
(#) inertia.phi - Absolute rotation angle of component	0	rad
(#) inertia.w - Absolute angular velocity of component (= der(phi))	0	rad/s
step.height - Height of step	1	null
step.offset - Offset of output signal y	0	null
step.startTime - Output y = offset for time < startTime	1	s
inertia.J - Moment of inertia	1	kg.m2
TqSD - Accelerating torque acting at flange (= -flange.tau) - start value	0	N.m
▾ Import parameters		
enable logging	no	
path to the unzipped FMU root	.../FMU_J1_Case1_Dymola_ME_J1	

図 3-3a　FMU1 (Model Exchange) パラメータ

Parameters

Title	Value	Unit
(#) speed.phi - Rotation angle of flange with respect to support	0	rad
spring.c - Spring constant	1	N.m/rad
spring.phi_rel0 - Unstretched spring angle	0	rad
damper.d - Damping constant	1	N.m.s/rad
fixed.phi0 - Fixed offset angle of housing	0	rad
speed.f_crit - if exact=false, critical frequency of filter to filter input s...	50	Hz
Nj1 - Reference angular velocity of flange with respect to support as i...	0	rad/s
▾ Import parameters		
enable logging	no	
path to the unzipped FMU root	.../FMU_SD1_Case1_Dymola_ME_SD1	

図 3-3b　FMU2 (Model Exchange) パラメータ

早速、FMU に設定されている Default のパラメータセットでシミュレーションを実行して結果を確認してみましょう。回転数 Nj1 とトルク TqSD は図 3-4 のような結果となったでしょうか？

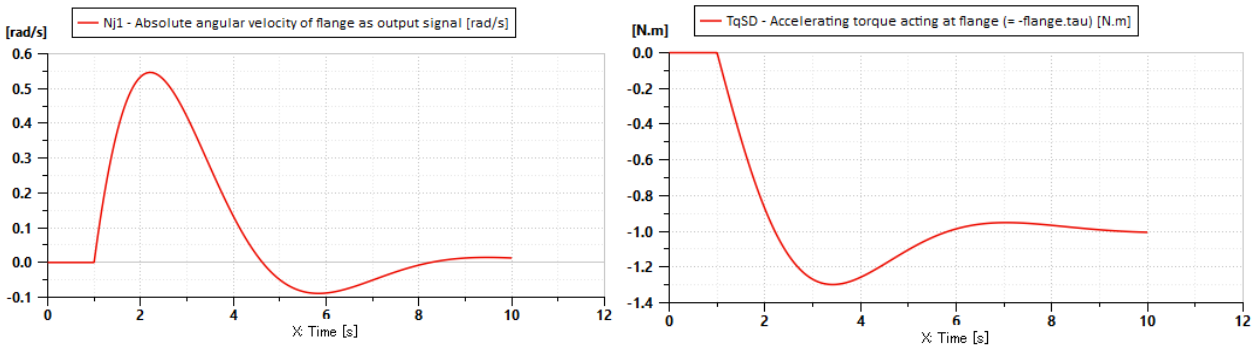


図 3-4　シミュレーション結果（Model Exchange）

3.1.2.2　　パラメータを変更して実行

次に、パラメータを変更してみましょう。共振周波数を上げ、ダンピングを低く設定します。

Parameters

Title	Value	Unit
speed.phi - Rotation angle of flange with respect to support	0	rad
spring.c - Spring constant	1	N.m/rad
spring.phi_rel0 - Unstretched spring angle	0	rad
damper.d - Damping constant	1	N.m.s/rad
fixed.phi0 - Fixed offset angle of housing	0	rad
speed.f_crit - if exact=false, critical frequency of filter to filter input s...	50	Hz
Nj1 - Reference angular velocity of flange with respect to support as i...	0	rad/s
Import parameters		
enable logging	no	
path to the unzipped FMU root	.../FMU_SD1_Case1_Dymola_ME_SD1	

図 3-5　FMU1 (Model Exchange) 変更前パラメータ

- バネ定数（spring.c - Spring constant）：
  現在の”1 [N･m/rad]”から 10 倍の”10 [N･m/rad]”へ変更
- ダンピング（damper.d - Damping constant）：
  現在の”1 [N･m･s/rad]”から 1/10 の”0.1 [N･m･s/rad]” へ変更

変更後シミュレーションを実行して、結果を確認してください。

回転数 Nj1 とトルク TqSD は図 3-6 のような結果となったでしょうか？

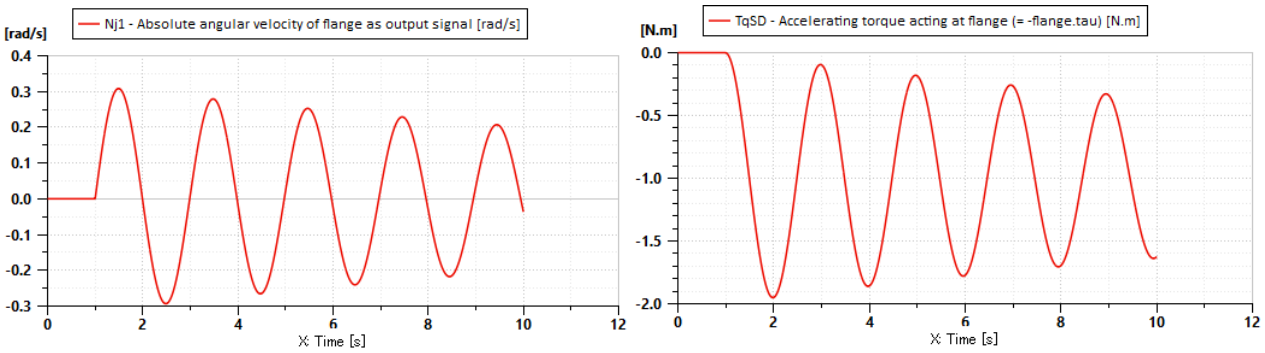


図 3-6　シミュレーション結果（ME：パラメータ変更後）

3.1.3　Co-Simulation でシミュレーションしよう

3.1.3.1　標準パラメータで実行

次に、Co-Simulation の FMU を取り込み、モデルを接続しパラメータを確認してみましょう。

パラメータの内容は、Model Exchange 時とほぼ同じですが、下記の 2 つが各 FMU で増えています。

- co-simulation step size
- co-simulation step specification

Parameters

Title	Value	Unit
# inertia.phi - Absolute rotation angle of component	0	rad
# inertia.w - Absolute angular velocity of component (= der(phi))	0	rad/s
step.height - Height of step	1	null
step.offset - Offset of output signal y	0	null
step.startTime - Output y = offset for time < startTime	1	s
inertia.J - Moment of inertia	1	kg.m2
TqSD - Accelerating torque acting at flange (= -flange.tau) - start value	0	N.m
▾ Import parameters		
co-simulation step size	0.001	s
co-simulation step specification	time step size	
enable logging	no	
path to the unzipped FMU root	.../FMU_J1_Case1_Dymola_CS_J1	

図 3-7a　FMU1 (Co-Simulation) パラメータ

Parameters

Title	Value	Unit
# speed.phi - Rotation angle of flange with respect to support	0	rad
spring.c - Spring constant	1	N.m/rad
spring.phi_rel0 - Unstretched spring angle	0	rad
damper.d - Damping constant	1	N.m.s/rad
fixed.phi0 - Fixed offset angle of housing	0	rad
speed.f_crit - if exact=false, critical frequency of filter to filter input s...	50	Hz
Nj1 - Reference angular velocity of flange with respect to support as i...	0	rad/s
▾ Import parameters		
co-simulation step size	0.001	s
co-simulation step specification	time step size	
enable logging	no	
path to the unzipped FMU root	.../FMU_SD1_Case1_Dymola_CS_SD1	

図 3-7b　FMU2 (Co-Simulation) パラメータ

これらは、Co-Simulation を行う際の通信間隔を設定するパラメータです。ここで表記されるパラメータの名前は、取り込み側のツールにより異なることがあります。今回は、”0.001 [s]”が通信間隔として設定されており、この間隔ごとに各 FMU への入力がやりとりされます。

それでは、FMU に設定されている Default のパラメータセットでシミュレーションを実行して結果を確認してみましょう。

回転数 Nj1 とトルク TqSD は図 3-8 のような結果となったでしょうか？

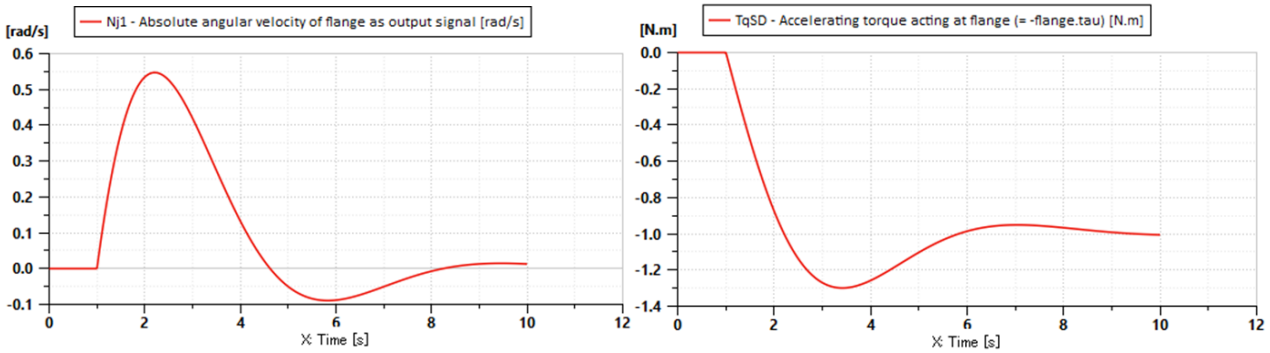


図 3-8　シミュレーション結果　(Co-Simulation)

Model Exchange 同様のシミュレーション結果がみられます。しかし、Co-Simulation を行う際には、通信間隔間では FMU の入出力は一定値として扱われ、Step Delay が生じます（2.4.2 参照）。この例では、0.001[s]という細かな通信間隔なので認識は困難です。出力結果のサンプリング設定を細かくして（0.0001[s]など）グラフを拡大してみましょう。図 3-9 のように通信間隔の 0.001[s]毎に結果が離散的に変化していることがわかります。

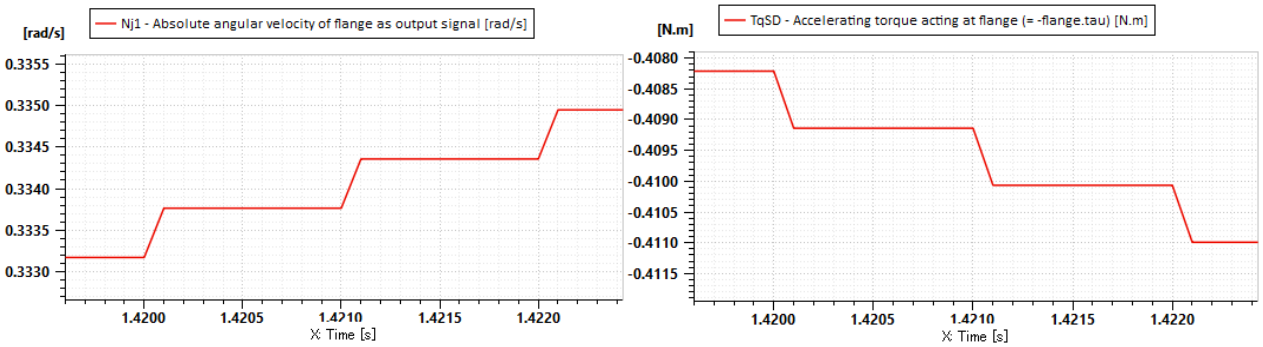


図 3-9　シミュレーション結果　(図 3-9 拡大)

3.1.2.3　　パラメータを変更して実行

次に、パラメータを変更してみましょう。FMU1、FMU2 の通信間隔を粗く設定します。

➢　通信間隔（co-simulation step size）：現在の”0.001 [s]”から 100 倍の”0.1 [s]”へ変更

通信間隔が粗くなり、離散的な結果が見えやすくなっています。また、通信間隔 0.001[s]の結果と比較すると、結果が異なっていることが見て取れます。Co-Simulation では、適切な通信間隔の設定が重要であることがわかります。詳細については 4 章で述べますので、そちらをご参照ください。

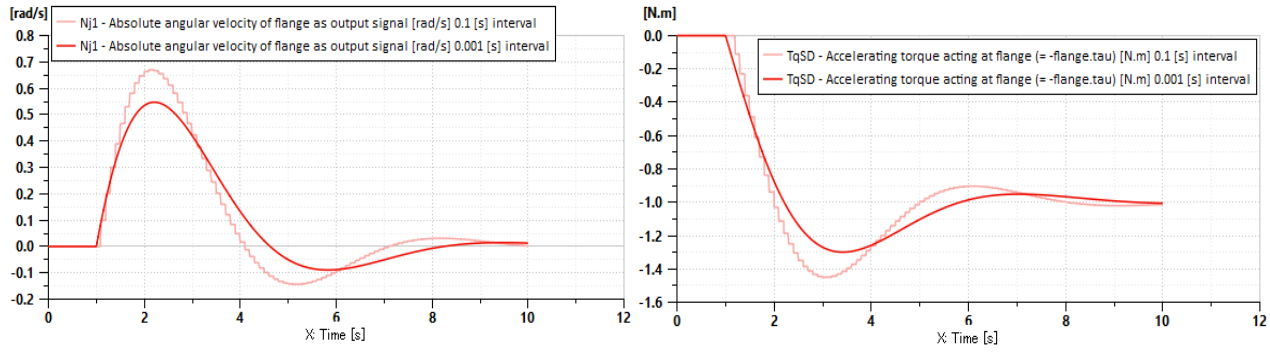


図 3-10　シミュレーション結果（CS：パラメータ変更前後の比較）

3.2　作ってみよう

3.2.1　サンプルモデルを使って FMU を生成してみよう

サンプルモデルは自動車技術会ホームページの自動車制御とモデル部門委員会ページに格納しています。モデルは、2018 年 5 月の自動車技術会春季大会にて、当 WG で講演した際のベンチマークモデルを使用します。[12]

ファイル名：Sample_3p2.zip　　：ZIP ファイルには以下のファイルが格納されています。

表 3-1　サンプルモデルの内容

	Model Exchange のオリジナルモデル		Co-Simulation のオリジナルモデル	
作成ツール	FMU1 用	FMU2 用	FMU1 用	FMU2 用
Amsim	J1_Amesim_ME.ame	SD1_Amesim_ME.ame	J1_Amesim_CS.ame	SD1_Amesim_CS.ame
Simplorer	J1_SD1_Simplorer_ME_CS.aedt　(1 つのモデル内に 4 種類全てが収納されています)			
Dymola	J1_Dymola_ME.mo	SD1_Dymola_ME.mo	J1_Dymola_CS.mo	SD1_Dymola_CS.mo
MapleSim	J1_MapleSim_ME.msim	SD1_MapleSim_ME.msim	J1_MapleSim_CS.msim	SD1_MapleSim_CS.msim
SimulationX	J1_SimulationX_ME.isx	SD1_SimulationX_ME.isx	J1_SimulationX_CS.isx	SD1_SimulationX_CS.isx
OpenModelica	J1_Modelica_ME.mo	SD1_Modelica_ME.mo	—	—

表 3-1 に対応するツールをお持ちであれば FMU を生成してみましょう。なお、FMU の生成方法は、使用するツールにより手順が異なりますので、詳細はツールのマニュアルをご参照下さい。

ここでは、Simcenter Amesim を用いた FMU の生成例をご紹介します。

まず､Co-Simulation のオリジナルモデルである FMU1 用の J1_Amesim_CS.ame を開いてください。図 3-11 の構成となりますが、このモデルはすでに FMI インターフェースブロックが挿入されており、FMU を生成する準備ができています。

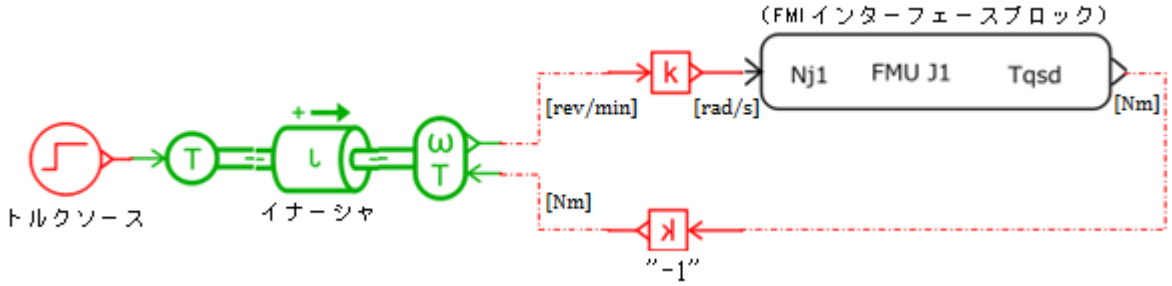


図 3-11　FMU1 出力用 J1_Amesim_CS.ame (J1 モデル)の構成

FMU の生成にあたり、入出力の単位系と信号の正負の向きに注意する必要があります。

モデル交換のルールは、経産省ガイドラインに基づいて決定しています（詳細 5.1.1 参照)。モデル間の受け渡しの単位は SI 組立単位系を採用するため、回転数は[rad/s]、トルクは[Nm]としています。

Amesim では回転数センサが[rev/min]を出力するので、"k"で単位変換して Nj1 を出力し FMU2 側へ出力します。

信号の正負の向きについては、エネルギソースからエネルギシンクへ流れる方向がエネルギの正の向きとなります。このモデルでは、トルクソース→イナーシャ→FMU2 側のバネ＋ダンパへの流れが正方向にあたります。FMU2 側から流入するトルクは、エネルギ伝達と逆方向に働くので、イナーシャが正

方向の回転となる場合には、回転数を低下させるため負方向のトルクが返ってくることになります。

Amesim のモデルでは、このトルクを正方向と定義しているため、”k”で－1 のゲインをかけて、正負の向きを反転するモデルにしています。この事例のトルクに限らず、物理量の正負の扱いはツールやモデルの作り方によって異なりますので注意が必要です。

モデルを FMU に出力するには、Interface > FMU Export Assistant を起動してください。図 3-12 のウィザードのメニューに従って FMU を生成します。

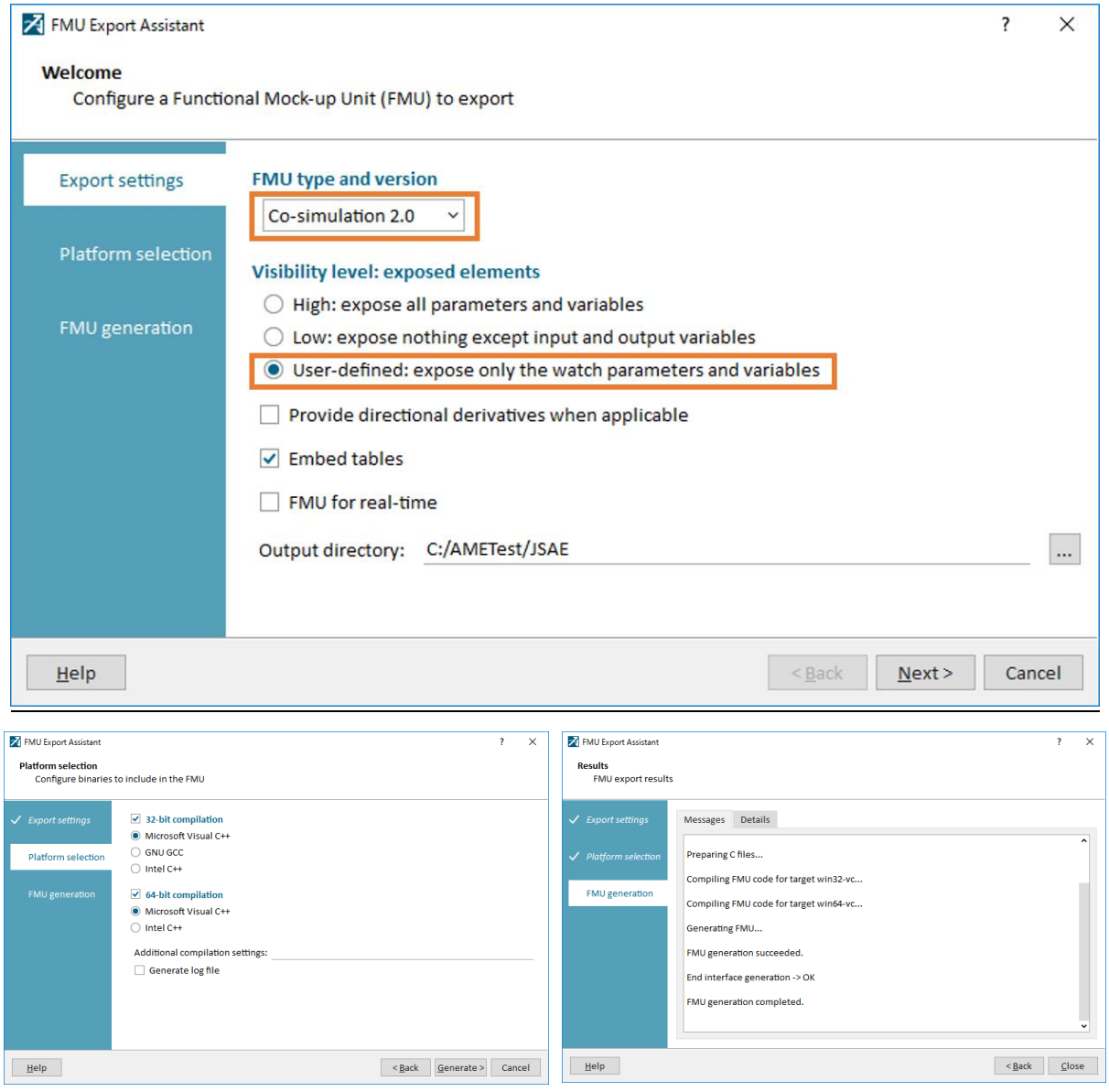


図 3-12　　FMU Export Assistant

今回は FMI 2.0 の Co-simulation を行いますので、該当する”FMU type and version”を選択します。

次にパラメータの表示レベルを決定するため”Visibility level: exposed elements”を設定しますが、”User-defined: expose only the watch parameters and variables”を選択すると、FMU に表示するパラメータや変数を任意に設定できるため、モデル流通後の FMU を利用する側にとって使い易いモデルと言えます。具体的な事例は、3.2.2 で説明します。

その後、使用するコンパイラーを選択して FMU を生成します。

FMU が生成できたら、再び取込んでシミュレーションを実行してみましょう。

3.2.2　必要なパラメータのみ表示しよう

FMU 生成時に、基となるモデルの作り方によっては、シミュレーション結果に影響を与えない不要なパラメータが表示されることがあります。不要なパラメータや変数が表示されると、FMU を利用する側の混乱を招くことになります。

ここでは、3.1.2 のサンプルモデルで使用した Model Exchange の FMU2 を例に説明します。図 3-13 のパラメータ”speed.f._crit – if exact=false, critical frequency of filter to filter input s…”は変更可能となって見えていますが、実際にはこのパラメータを変更してもシミュレーション結果に影響を与えません。何故このようなことが起こるのでしょうか。

Parameters

Title	Value	Unit
speed.phi - Rotation angle of flange with respect to support	0	rad
spring.c - Spring constant	1	N.m/rad
spring.phi_rel0 - Unstretched spring angle	0	rad
damper.d - Damping constant	1	N.m.s/rad
fixed.phi0 - Fixed offset angle of housing	0	rad
speed.f_crit - if exact=false, critical frequency of filter to filter input signal	50	Hz
Nj1 - Reference angular velocity of flange with respect to support as input signal - start value	0	rad/s
Import parameters		
enable logging	no	
path to the unzipped FMU root	../FMU_SD1_Case1_Dymola_ME_SD1	

図 3-13　FMU2 (Model Exchange) パラメータ

FMU2 を生成する前のオリジナルモデルは Modelica で作成しており、図 3-14 の構成となっています。対応するツールをお持ちであれば SD1_Dymola_ME.mo（SD1 モデル）を開いてみましょう。

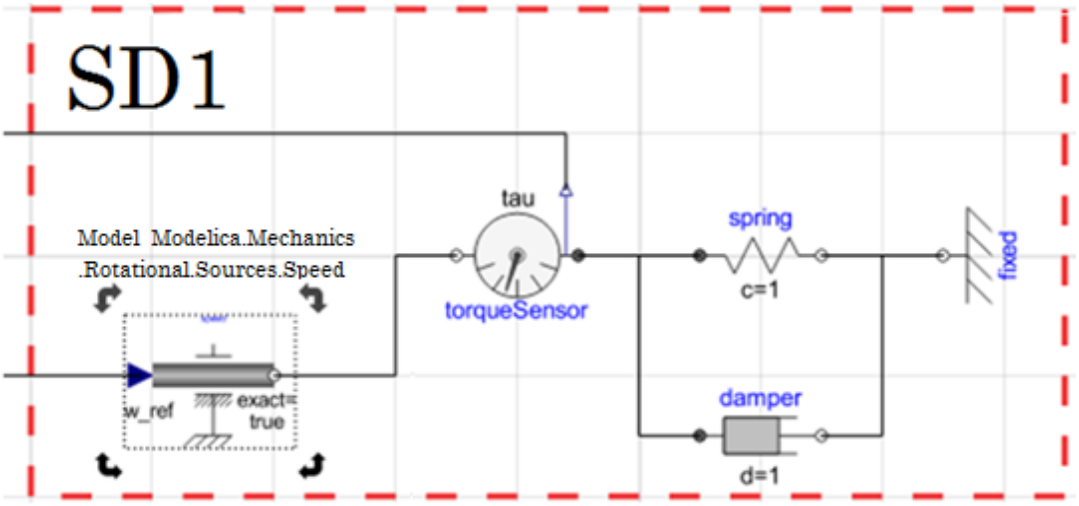


図 3-14　オリジナルの Modelica (SD1 モデル)の構成

このモデルでは“Model Modelica.Mechanics.Rotational.Sources.Speed“を使っており、このブロックのパラメータが図 3-15 になります。

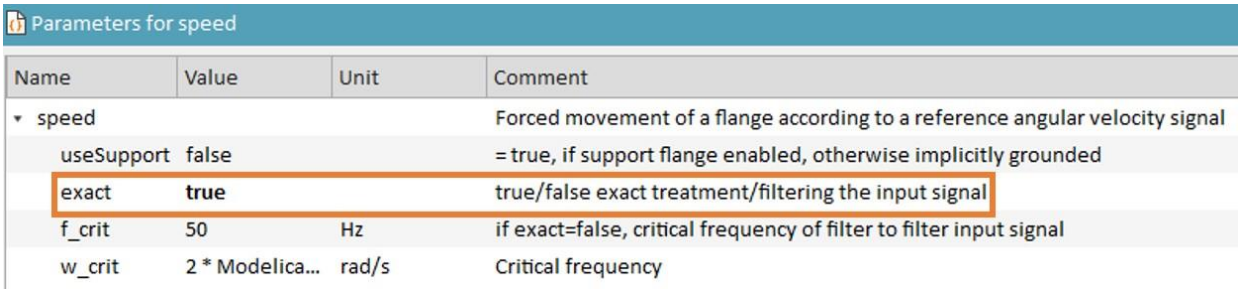


図 3-15　(SD1 モデル) speed のパラメータ

パラメータに”f_crit”がありますが、”exact”が false のときのみに有効になります。FMU 生成時に”exact”を”true”で設定すると、FMU 生成後は変更ができません。

したがって、図 3-13 で”speed.f._crit – if exact=false, critical frequency of filter to filter input s…”の値を変更しても無効となります。

そもそも FMU 生成する際に”f_crit”を変更可能なパラメータとして扱ったために起こる現象なので、必要なパラメータや変数のみ表示できるように設定し、変更の必要がないパラメータは隠蔽することが、利用者の混乱を防ぐことに繋がります。

第 4 章　FMU の動作モード（ME/CS）選択のガイドと各工程の流れ

この章では、Model Exchange(ME)と Co-Simulation(CS)についてもう少し詳しく説明します。
FMI には ME と CS の二つの方式があり、どちらを使うべきか悩まれると思います。4.1 では、それぞれの特徴を知ることで適した手法を選択する指針を解説しています。4.2 では、モデル作成からシミュレーション実行までの工程について、4.3 及び 4.4 では、トラブルが発生した際の問題解決への糸口を見つけるため、それぞれの工程で起きうる問題と対策について解説しています。

4.1　FMU の動作モード（ME/CS）選択の指針

表 4-1 に、Model Exchange、Co-Simulation それぞれの特徴を表 4-1 にまとめました。表の最後に、当ワーキンググループ推奨の用途をまとめました。Model Exchange が有用な場合としては、制御モデルの交換が挙げられます。その理由としては、制御モデルは一般的に代数ループを含まない点、通信遅延が無い事が望まれる点、モデルが一般的に陽解法固定タイムステップソルバで解け、ソルバとモデルの相性問題が起こりにくい点が挙げられます。Co-Simulation を用いる場合は、それぞれ適したツールでモデルとソルバの相性が確認された FMU 同士をつなげることが可能であるため、複合物理システムモデルに有用と言えます。また、各 FMU が個別のプロセスを持つことが可能なため、並列計算を行いシミュレーションの計算速度向上を図る事も期待されますし、マルチレートでの実行も可能です。
Hardware in the Loop Simulation を行う際にも、モデル作成ツールや、実行ハードウェアに依存しない流通が可能です。

表 4-1　ME／CS の特徴

	Model Exchange	Co-Simulation
代数ループを含んだ FMU	FMU 生成不可能、もしくは安定性が損なわれるため、非推奨	生成元ツールのソルバにより安定した計算が可能 計算時間増大の可能性
通信遅延	発生無し	Communication Step Size の適切な設定が必要
可変タイムステップソルバ使用時	モデル／ソルバ相性の確認が必要	モデル／ソルバ相性は FMU 生成時に確認済
陽解法固定タイムステップソルバ使用時	モデル最大固有値の抑制が必要	モデル最大固有値の抑制が必要
Core 分割	不可能 モデル全体で単一プロセス	可能 FMU 毎にプロセスを生成可能
推奨用途	制御モデル	複合物理システム 大規模モデルの並列計算処理 マルチレート Hardware in the Loop Simulation

4.2　FMU 作成から、交換、シミュレーション実行までの流れ

シミュレーションを成功させる上で、トラブル時の原因究明を行うには、どのような流れで FMU 生成から、交換、シミュレーション実行が行われているかの理解が役に立ちます。大まかな流れを、図 4-1 にまとめます。

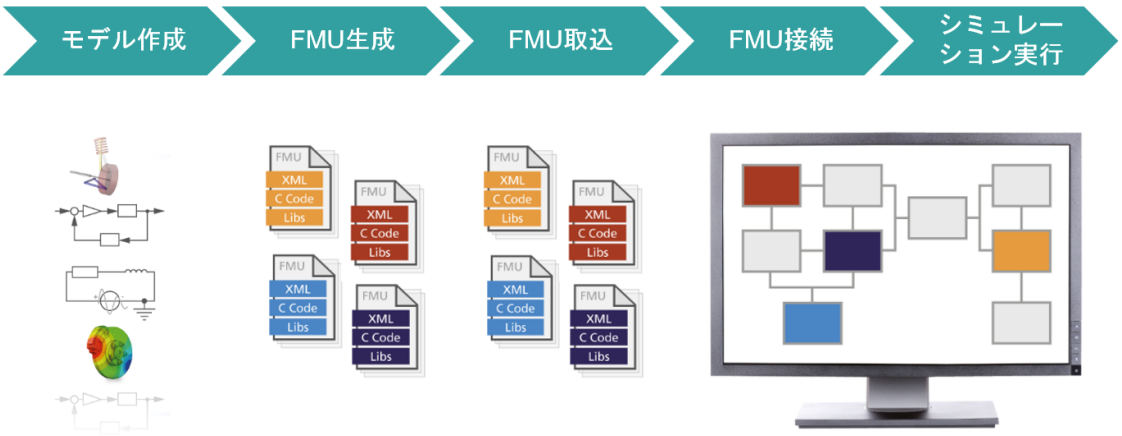


図 4-1　FMU 生成から、交換、シミュレーション実行までの流れ
(FMI ホームページより引用、一部修正) [10]

4.3　それぞれの工程で起こりうる問題と対策（Model Exchange）

4.3.1　モデル作成時

Model Exchange にて、FMU を作成するには出力するモデル内部に代数ループが無い事が推奨されます。作成したモデルに代数ループがある場合は、その原因特定と対策を行い陽的なモデルに変更しましょう。

例を見てみましょう。図 4-2a のモデルでは、可変のオリフィスを PI 制御で目標の流量に達成するように開口径を変化させていますが、図 4-2b のハッチング部に代数ループが発生しています。

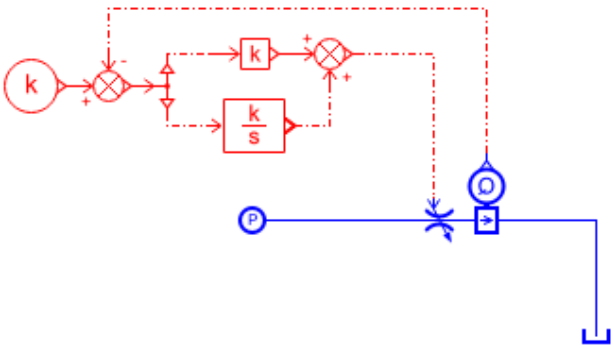


図 4-2a　流量の PI 制御例

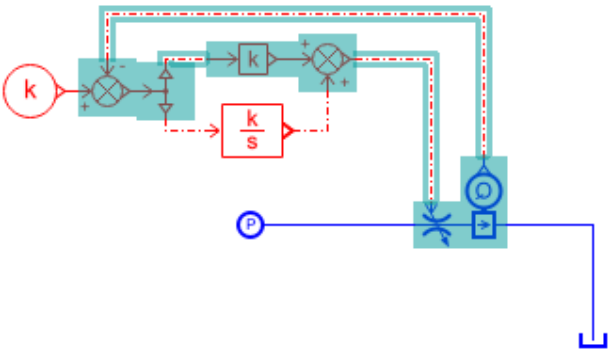


図 4-2b　流量の PI 制御例 代数ループ発生個所

図 4-3 のように、一巡ループ間に積分計算を挿入することで代数ループを解消することが出来ます。Actuator の時間応答遅れを模擬し、出力に 1 次遅れを挿入することが一つの対策として考えられます。

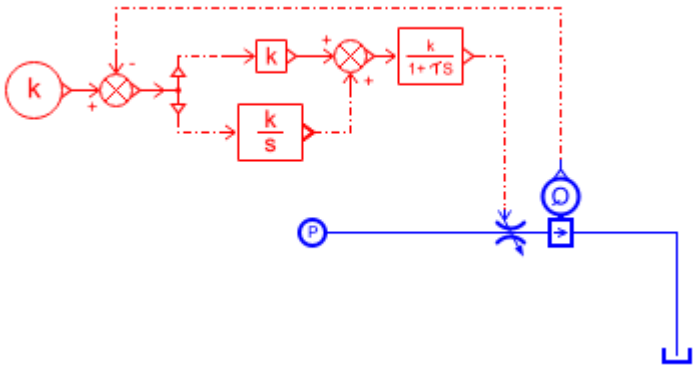


図 4-3　1 次遅れを挿入

このように物理モデル内に PI 制御を用いている場合に発生することがある点に注意しましょう。また、物理モデルのみで代数ループが発生する場合もあります。電位一定、変位一定などの拘束条件を持つ計算素子を用いた場合は FMU 生成前に代数ループの有無を確認しましょう。

4.3.2　FMU 生成／取込時

FMU のパラメータや状態変数名、入出力の情報などは、model Description File（modelDescription.xml）に格納されています。これらの情報を正しく書き出し、読めるかどうかが FMU 交換成功の鍵です。FMI Specification にどのように model Description File を記述すべきかが書かれており、また Compliance Checker が FMI ホームページにて公開されており、FMU 出力に対応しているツールベンダはこのチェッカにて標準対応しているかを検証しています。

4.3.3　FMU 接続時

FMU の入出力を正しく接続できるかが、最初の関門です。3 章での例題では、2 つの FMU 間の接続は 1 つの組み合わせしかなく、問題なく接続ができました。しかし、複数の入出力がある FMU を取り扱う場合は、どの入出力を接続するか確認する必要があります。様々な制御や物理モデルをやり取りする場合に、FMU 間でやり取りすべき変数は何か、また単位はどうすべきかを判断するには、参考例として、経済産業省が発行した「自動車開発におけるプラントモデル I/F ガイドライン（ver1.0）」［11］が活用できます。これらを元に入出力ルールを、あらかじめモデル交換者間で合意する必要あります。この点については、5 章で詳しく述べます。

Model Exchange 接続時に発生しうる問題としては、FMU を接続し、入出力の繋がりで循環がある場合、本当は存在しない不必要な代数ループが発生することがあります。その場合、model Description File の Dependency をユーザにて変更し、代数ループの解消を行う事も考えられます。

具体例を見てみましょう。これは、3.1.2 の例題にて、回転慣性側（FMU1）として生成された model Description File 例です。

```
<ModelStructure>
    <Outputs>
        <Unknown index="9"/> ⟶ Output signal (speed)
    </Outputs>
    <Derivatives>
        <Unknown index="2"/> ⟶ Acceleration (derivative of speed)
        <Unknown index="4"/> ⟶ Speed (derivative of displacement)
    </Derivatives>
    <InitialUnknowns>
        <Unknown index="1"/> ⟶ Speed
        <Unknown index="3"/> ⟶ Displacement
        <Unknown index="9"/> ⟶ Output signal (speed)
    </InitialUnknowns>
</ModelStructure>
```

FMU 内部で取り扱われる変数には、index が設定されています。積分計算される状態変数および、その他 FMU 生成元モデル上で使用されている変数に対してこの index が設定されます。このケースでは出力される信号は回転数（index=”9”）で、状態変数（index=”1”）の値を検出したものです。

これら 2 つの変数の依存性を宣言されていれば、取り込み側ツールは、出力信号が積分計算される状態変数であることがわかり、入出力の関係性が代数方程式では無い事がわかります。しかし、上記の例では依存性が表現されていないので、取り込み側のツールが、入出力関係が代数方程式である可能性があると理解し、同様の理解を FMU2 側に行えば、FMU1 と FMU2 間で代数方程式のループが生じると理解されます。実際には、各 FMU の出力値は入力信号を積分計算された値であり、陽解法ソルバを使って解けるはずなのですが、代数ループを解くことが出来る陰解法ソルバを使用することとなります。

陰解法を用いて解く場合は、一巡伝達関数ゲインが 1 以下であることが求められます。

この例にて代数ループを解消するには、下記のように model Description File を変更することが一つの対策例です。

```
<ModelStructure>
    <Outputs>
        <Unknown index="9" dependencies = "1"/>
    </Outputs>
    <Derivatives>
        <Unknown index="2"/>
        <Unknown index="4"/>
    </Derivatives>
    <InitialUnknowns>
        <Unknown index="1"/>
        <Unknown index="3"/>
        <Unknown index="9"/>
    </InitialUnknowns>
</ModelStructure>
```

“dependencies”を宣言することで、出力値（index=”9”）が積分して解かれる状態変数であることが理解でき、不必要な代数ループの発生を防ぐことが出来ます。

上記の例のような対策が必要かどうか、もしくは代数ループ発生時の解消対策可能かどうかは、ツールに依存することをご留意ください。代数ループに関する問題が起きた際の対策の一例としてご紹介しています。

4.3.4　シミュレーション実行時

Model Exchange では、取込側のツールのソルバを用いて計算を行います。シミュレーションがうまく流れるかは、モデルとソルバの相性が良いかと言い換えることができます。この相性を理解するには、シミュレーションの安定性に関する知見が役に立ちます。ここでは、簡単に技術的内容をご紹介します。

FMI を用いて交換されるモデルは、主に時間に関する常微分方程式のみで表されています。

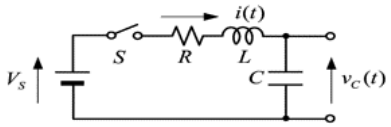


図 4-4　LRC 回路

$$L\frac{d^2q(t)}{dt^2} + R\frac{dq(t)}{dt} + \frac{1}{C}q(t) = V_S \qquad (1)$$

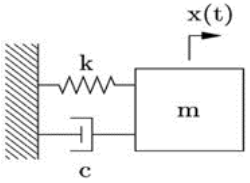


図 4-5　単振動システム

$$M\frac{d^2x(t)}{dt^2} + C\frac{dx(t)}{dt} + Kx(t) = F_{ext} \qquad (2)$$

このように様々な物理領域において、モデル化対象の振る舞いが式(1)、式(2)のように 2 階の常微分方程式で表されます。この形は LRC 回路（図 4-4）や単振動システム（図 4-5）のように固有値や共振周波数を持ちます。解いているモデルの固有値が使用しているソルバの収束領域に入っているということが、発散しない、解けるシミュレーションが行われている状態です。参考までに、Runge-Kutta ソルバの安定領域を図 4-6 に示します。複素平面上に表した領域に固有値がある事が収束の条件です。領域は、ソルバタイムステップのΔt によって変化します。高固有値（高い周波数もしくは、短時定数）が存在する場合は、Δt を細かくして収束領域の中に収める必要があります。

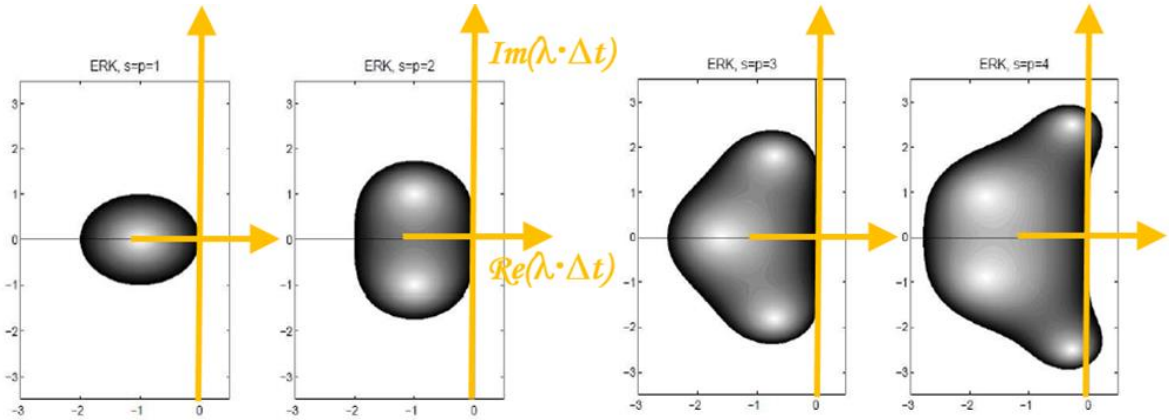


図 4-6　Runge-Kutta ソルバの安定領域（1～4 次）

詳細な物理モデルを取り扱うと、線形時不変ではなく非線形なシステムであることがほとんどです。その際は、適切な時間刻みやソルバ次数などを変更しながら解いていく、可変タイムステップソルバが用いられることがあります。可変タイムステップソルバは、シミュレーションツールそれぞれで工夫がなされており、ツールで作成されるモデルを解きやすいようにチューニングされています。その為、あるツールで作成した FMU を Model Exchange で出力し、他ツールで取込み、シミュレーション実行する際にうまく収束しない、もしくは結果に差異が生じるなどの問題が発生する可能性があります。これが、モデルとソルバの相性の問題です。

4.4　それぞれの工程で起こりうる問題と対策（Co-Simulation）

4.4.1　モデル作成時

Co-Simulation では、モデルとソルバを合わせて出力します。そのため、仮にモデル内で代数ループがあったとしても問題ありません。拘束方程式を使用した場合の機構や電気回路モデル、もしくは 3D-CFD などの特殊なソルバが必要なモデルを使用する際に活用できます。

4.4.2　FMU 生成／取込時

FMU のパラメータや状態変数名、入出力の情報がうまくやり取りできるかは、Model Exchange と同様の問題を持ちます。

4.4.3　FMU 接続時

FMU 入出力の整合は、Model Exchange と同様ですが、Co-Simulation 時に不必要な代数ループが発生することはあまりありません。なぜならば、Communication Step Size による通信遅れがあるために代数ループが解消されるからです。ほとんどの取り込み側ツールでは、Co-Simulation 使用時には FMU を接続し、入出力の繋がりで循環がある場合でも、代数ループとみなさず処理が行われます。

4.4.4　シミュレーション実行時

Co-Simulation を行う際は、3 通りの方法があります。1 つ目は、ソルバとモデルを合わせて出力する Coupling with System Models もしくは単純に Stand Alone と呼ばれる手法です。2 つ目及び 3 つ目は Tool Coupling、Distributed と呼ばれているもので、ツール間のインターフェースの役割のみを果たし、双方のソフトウェアを立ち上げて行う Co-Simulation です。手法の違いによって、実行時に必要なソフトやインストール環境が異なる点に注意してください（2.4.1 参照）。

計算結果において、発散する、もしくは計算速度が遅くなる場合があります。Co-Simulation での Communication Step Size により生じる問題です。また、シミュレーション結果はこの Communication Step Size 設定によって差異が生じます。

Co-Simulation では、マスタ、スレーブ間で決められた時間間隔で通信を行います。通信間隔の間は、各 FMU の入力値が一定と扱われるため、ステップ応答遅れが FMU 間に生じます。このステップ応答遅れが挿入されることで下記のような課題が起こりえます。（一部のツールでは、連続的な入力となるように補間を行う場合もあります。）

1) 通信遅れにより、接続したモデル系全体での安定性が損なわれ、発散リスクが高まる。通信間隔が大きい程問題となるため、この視点からは、細かい間隔が望ましいです。
2) また、通信間隔を小さくすればするほど、ステップ応答遅れが及ぼすモデル全体系への影響も小さくなるため、計算誤差の低下も期待できます。
3) しかし、細かい通信間隔を設定すると通信時の不連続点が増大し、ソルバでの収束演算回数が増すことで計算時間が増大する。

通信間隔をなるべく大きくとれるよう、モデル間の連成が弱い点での接続を行うことが望ましいです。また収束計算演算数を抑えるために、陽解法固定タイムステップソルバを使うなどの対策が有効ですが、そのモデルの固有値がソルバ収束領域内にすべて入っており、発散しないことを確認する必要があります。

なお、利用するツールによって機能の有無がありますが、PC に複数コアまたはスレッドがある場合、Co-Simulation では FMU 毎に別個の executable を実行できるので分散処理が可能です。これに対し Model Exchange では、基本的にモデル全体で 1 つの executable を実行するので、分散処理には向いていません。この機能をうまく活用すれば計算速度の向上が期待出来ます。

第 5 章　実務適用のために知っておきたいこと

この章では、FMI を実際に使うために FMU 生成上の注意点から読込実行までのエラーなどについて説明を行います。5.1 では、プラントモデルの入出力 I/F を決定する上で指針としている経産省ガイドラインについて解説します。5.2 では、様々な部類のモデルを分割する際の注意点を挙げています。5.3 では、FMU の互換性及び隠蔽性について説明します。5.4 では、シミュレーションツールの違いによる注意点を挙げています。5.5 では、FMI を利用する上で起こりうるエラーについて解説しています。尚、一般のツール連携における注意点と FMI 特有の注意点を区別するため、【共通】、【FMI】を表記しています。

5.1　プラントモデルの入出力決定の指針

複数のモデルをひとつの全体モデルにまとめる場合、モデルの入出力関係が重要になります。FMI を使用した場合も同じです。5.2 ではプラントとコントローラを 4 つの関係に分けて接続・分割についての注意点を説明しますが、その前に 2017 年 3 月に経済産業省が発表した「自動車開発におけるプラントモデル I/F ガイドライン（以下、経産省ガイドライン）」 [11] を確認しておきます。ここではまず提示された 5 つの原則を簡単に説明し、この原則と FMU 化の対象となるモデルの入出力について説明します。

5.1.1　経産省ガイドラインの考え方【共通】

プラントモデル間の物理的なつながりを表す物理量の授受は、必ず「出力」「入力」をセットにして考える必要があります。機械（回転）系ではトルクと角速度（角度の場合もある）、電気系では電圧と電流になります。経産省ガイドラインでは入出力信号の組み合わせ、符号、単位系を定めるために 5 つの原則を挙げています。

表 5-1　経産省ガイドラインの原則

基本原則

第一	プラントモデル間はアクロス変数とスルー変数でつなぐ。 また、アクロス変数とスルー変数の向きは互いに逆向きとする。
第二	エネルギソースからエネルギシンクへ流れる方向をエネルギの正の向きとする。
第三	スルー量・アクロス量を蓄積する要素を基準として、全体の I/F を考える。
第四	スルー変数の正負は、エネルギの正の流れの向きと、スルー変数の入出力の向きが同じとき、正とする。
第五	入出力の単位は SI 単位系、SI 組立単位系を利用する。 量記号は JIS 規格を使用する。

（自動車開発におけるプラントモデル I/F ガイドライン(ver.1.0)より引用)[11]

FMU 生成における重要な点と経産省ガイドラインの原則を関係づけながら説明します。なおスルー量、アクロス量については経産省ガイドラインを参照して下さい。

本活用ガイドでも変数の種類など同じ扱いをしています。

5.1.2　接続信号の選択【共通】

経産省ガイドラインでは第一原則、第三原則の適用部分です。第一原則によれば、プラント間の接続では、図 5-1 の様に一方のプラントがスルー量入力の場合、そのプラントはアクロス量を出力します。接続先のもう一方のプラントはそのアクロス量を入力として受け取り、はじめのプラントが受け取るスルー量を出力することになります。第三原則では表 5-2a、5-2b のように接続部位において表現されている物理的な部品を 2 つに分けて考えます。

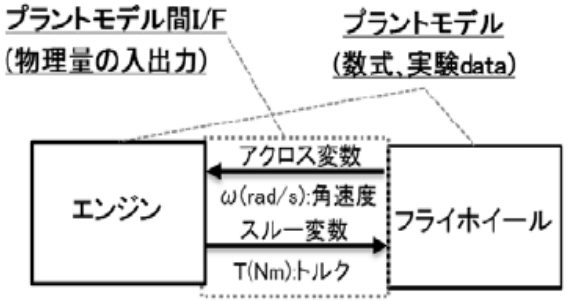


図 5-1 第一原則　（経産省ガイドラインより引用）[11]

表 5-2a　スルー量を蓄積する要素（スルー変数を入力とする）

物理領域	機械（並進）	機械（回転）	電気	熱
物理的な部品	質量	慣性モーメント	電気容量（キャパシタ）	熱容量
入力：スルー変数	力	トルク	電流	熱流量
出力:アクロス変数	速度	角速度	電圧	温度

表 5-2b　アクロス量を蓄積する要素（アクロス変数を入力とする）

物理領域	機械（並進）	機械（回転）	電気	熱
物理的な部品	バネ	ねじりバネ	電気誘導（インダクタ）	該当なし
入力：アクロス変数	速度	角速度	電圧	該当なし
出力：スルー変数	力	トルク	電流	該当なし

例として機械（回転）で考えてみると次のようになります。
慣性モーメントはスルー量（トルク）を蓄積します（表 5-2a）。このため慣性モーメントがモデルの接続部についている場合は、トルクが入力で角速度（アクロス量）を出力とすることが推奨されています。非因果ツールで速度入力の FMU を生成しようとした場合には生成時にエラーが発生し、回避するためには加速度入力に変更する必要があります。しかし加速度を生成するためには微分が必要になり数値シミュレーション的に不安定になりやすいと考えられています。このため経産省ガイドラインではアクロス変数ではなくスルー変数を入力することを原則としています。なおプラントモデルが非因果モデリングツールを用いて作成されている場合、自動車技術会「非因果モデリングツールを用いた FMI モデル接続ガイドライン Ver.1.0」[4] で説明されている非因果アダプタを適切に用いればこの原則とは異なる接続を実現することができる場合もあります。

経産省ガイドラインに基づく接続では、慣性モーメント同士の接続ができません。この場合、慣性モーメントをどちらかにまとめるのが好ましいと当 WG では考えます。やむを得ず両側に慣性モーメントを持つ形で分ける場合には、いずれかに回転バネ要素を入れることでスルー変数入力からアクロス変数入

力に切り替える必要が生じてしまいます。この場合、回転バネと慣性モーメントの大きさを適切な関係にしないと、小さな時定数を持つシミュレーション時間が長い不安定なモデルになってしまいます。

ねじりバネが接続部にある要素ではこれとは逆にアクロス量（角速度）を入力することができます。スルー変数であるトルクを出力とすることができます。

なお経産省ガイドラインでは、減衰（ねじりダンパ）が接続部にある場合は、スルー変数入力でもアクロス変数入力でも構わない、としています。機械系では実際に減衰のみを接続部に持つことはあまり見受けられません。電気系では電気抵抗がスルー変数入力・アクロス変数入力の両方に対応し、これは十分に考えられる接続です。

5.1.3　信号の取り決め【共通】

5.1.3.1　入出力の符号

第二原則、第四原則が適用される部分です。経産省ガイドラインでは第二原則の例として、エンジン側から駆動系、走行抵抗に向かうエネルギの流れ（仕事率、またはパワー）を正と定義しています。これに従うと変速機からデフへトルクを入力する（デフから変速機へ角速度を入力する）という関係は、加速時に正のトルクをデフへ送る、減速時には負のトルクをデフへ送るという関係になります。

5.1.3.2　単位系

第五原則で単位系は SI 単位系としています。FMI では modelDescription.xml に単位系を記載し内部で明示的に持つことができます。しかし FMI の規格において単位系を利用する機能は必須項目ではないため、生成された FMU 内に必ずしもこの記述があるとは限りません。また取り込み側でその単位系定義を使用しているとも限りません。

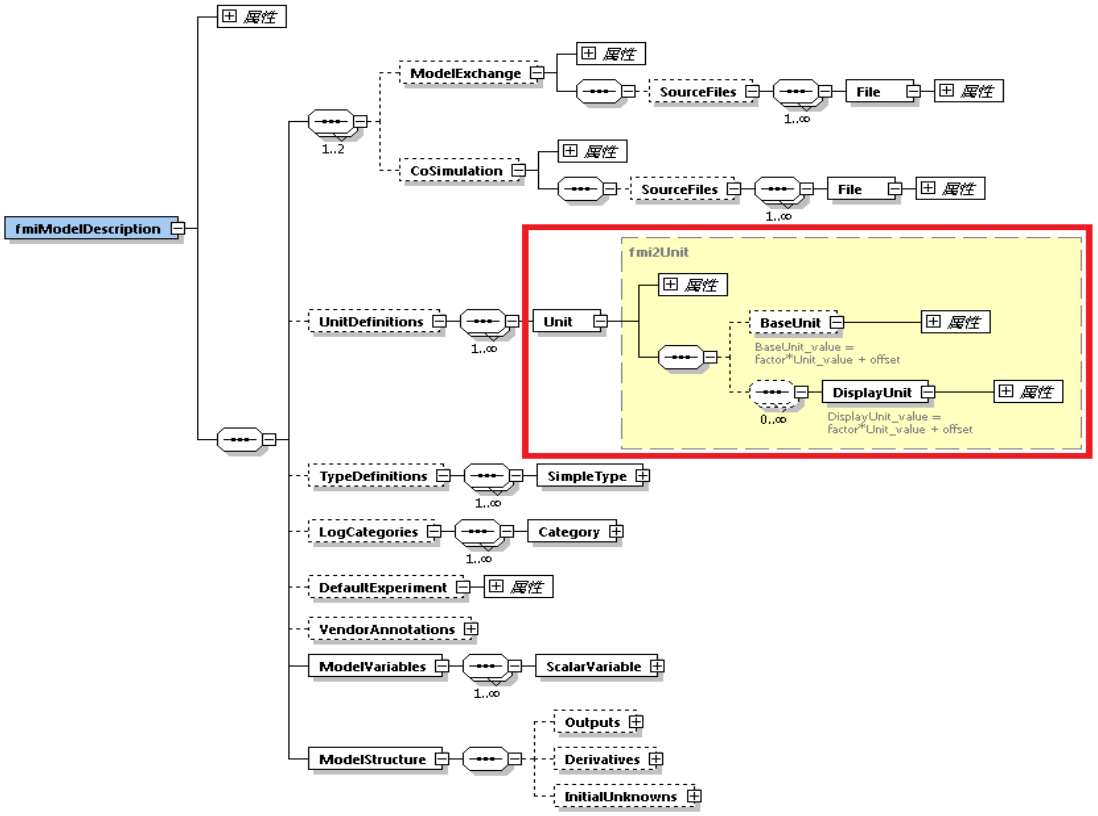


図 5-2 modelDescriptionFile の構造（対象部分のみ構成展開；赤色枠内）
（FMI 仕様書：FMI for ModelExchange and CoSimulation v2.0 より引用）[7]

自動車開発においては、回転数（回転速度）を表すのに[rad/s]ではなく [rpm]を単位として用いることが多く見受けられます。またしばしば温度は摂氏[℃]で表現されます。このように単位系は誤解を招きやすいため、経産省ガイドラインではモデルそのものとは別情報の「サブシステム I/F 定義書」として授受することが提案されています。当 WG でもモデルである FMU の外で伝達することが望ましいと考えます。

5.2　モデルの分割における注意点

前節でプラントモデルの分割について触れました。ここではサブモデル作成上について 2 つの接続先をコントローラとプラントに分けて分割の説明をします。

5.2.1　コントローラとコントローラ【FMI】

アナログ（連続系）信号入出力とデジタル（離散系）信号入出力を分けて考える必要があります。ME を用いた場合、実際のアナログ信号はモデル間で遅れを伴わずシミュレーションが行われます。CS を用いた場合は、信号授受で通信間隔分の遅れが生じることを配慮しなければなりません。デジタル信号ではもともとサンプリング分の遅れがあるので、その遅れと CS の通信間隔との関係から影響の有無を考察しなければなりません。

推奨（CS の場合）

1 つのコントローラで行われる処理は FMU を分けずに 1 つの FMU の中で行う。

通過するだけの信号は FMU を経由して送信しない。できる限り信号源となるコントローラと受信側コントローラを直接結ぶ。

FMI2.0 まではバス（またはベクトル）端子は存在していません（FMI3.0 で改善予定）。コントローラ間を接続する場合、単純にバスのような接続を行うと多数の接続が発生してしまいます。必要な入出力に限定して接続を行うようにして下さい。特に入力端子については入力信号がないとエラーになるツールも多いので不要な端子を設けるべきではありません。2 章（図 2-5、2-10）に示されるように、FMI の規約では出力変数(y) 以外に開示変数(v) が定義されています。読み込んだツール上で観測したい変数がある場合には FMU を生成する時点で(y) または(v) として定義する必要があります。一方、読込側ツールによっては(v) を観測できないものもあります。この場合、接続をしない変数も出力変数として設定を必要とする場合があります。

5.2.2　コントローラとプラント【共通】

5.2.2.1　コントローラとプラント間の信号

一般にコントローラとプラント間での送受信される指示値や観測値信号はケーブルを通じた電気信号ですが、回路としてモデル化されるケースは多くありません。厳密にいうと電気回路における信号の遅延を考える必要があります。どの部分でアクチュエータへの指示信号やプラントが出力するセンサ信号が離散化されるのかを考えてコントローラとプラントを分割する必要があります。特に CS の場合、前項でも取り上げたように遅延が正しく表現されているかを考える必要があります。

5.2.2.2　コントローラとプラント間の物理量

モデルの上では 5.2.2.1 で書いた信号と似ていますが、スルー変数、アクロス変数として実際に接続されている物理的な値のやりとりをする場合は 5.2.3 で示すプラントとプラントの接続と同じ状態になります。このため 5.2.2.1 で書いたような信号と同じなのかを判断する必要があります。

例えば図 5-3a のようにコントローラ（FMU1）からプラント（FMU2）のモータへの電流値を伝え、プラントから電圧が戻るような場合は、作成者側は FMU1 をコントローラとして考えたとしても、コントローラ自身にプラントの特性を持つため、結果は図 5-3b のような挙動を示します。

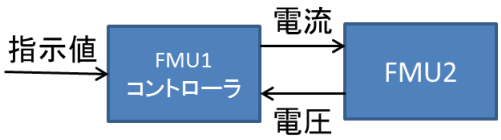


図 5-3a　コントローラとプラントのモデル例

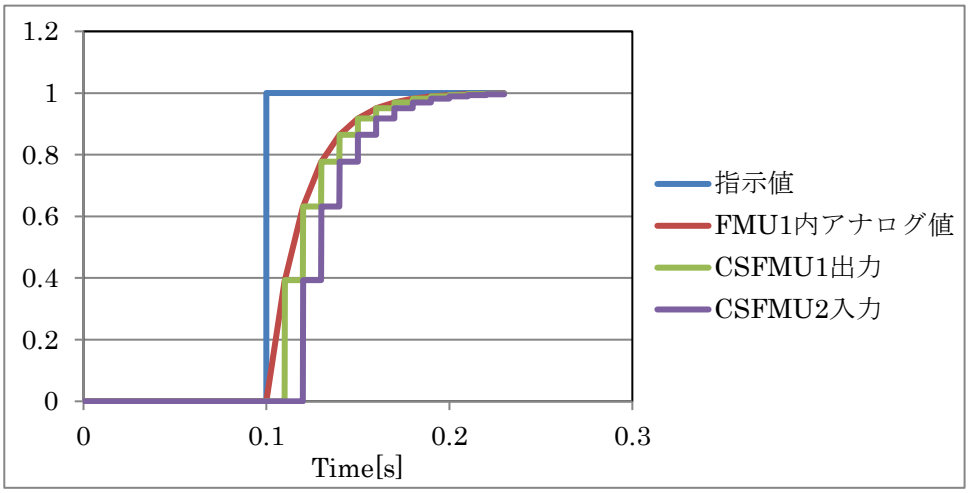


図 5-3b　CS における遅延の例　(通信間隔=0.01s)

5.2.3　プラントとプラント【共通】

もっとも注意を払うべきなのはプラントとプラントの接続で機械的、電気的など物理的な接合を異なるモデルに分割する場合です。入出力については経産省ガイドラインに沿って考えることを当 WG では推奨します。FMU を読み込んだツールが非因果接続を認めるツールである場合には、5.1.2 で触れた非因果アダプタを用いる方法もあります。5.1 で説明したように経産省ガイドラインに従って分割・接続をする場合、スルー変数を入力とする部品同士、アクロス変数を入力する部品同士を直接接続することができません。

5.2.3.1　モデル分割を避けたい位置

動作が速い（時定数が小さい）部位や、元々の動作の不連続性が高い部位での分割も可能な限り避けるべきと考えます。

・時定数が小さい部位の例：機械系では質量／慣性モーメントに対して並進／回転バネ定数が大きい。
・不連続性の高い部位の例：機械系では摩擦や突き当たり（衝突、機構的な可動範囲の制約）、電気系ではダイオードなどのスイッチング素子が該当します。

5.3　互換性と隠蔽性【FMI】

5.3.1 で述べるように FMI は仕様決定時期によりバージョン 1.0 と 2.0 がありますがそれぞれの互換性はありません（2.2.2.1 参照）。また 5.2.3 で説明するように FMU を実行する環境により生成する FMU も異なってきます。

同一モデリング言語レベルでモデル交換を行う場合、ノウハウの詰まったモデルそのものを交換することになり、内部構成を開示してしまうことになります。FMI ではモデル内部を隠蔽して FMU を提供できる点が利点として挙げられます。5.3.3 でその注意点を説明します。

5.3.1　FMI バージョン互換性

FMI は 1.0（1.0.1）と 2.0 が存在していますが上位互換性はありません。このため FMU を読込み実行する環境で 1.0 と 2.0 のいずれに対応しているのかを作成する側が確認して提供する必要があります。両方のバージョンを混在させて実行できる環境も多く存在しますので、2.0 で接続可能な部分はできるだけ 2.0 を用いて下さい。

5.3.2　OS 互換性

FMU 内部には「binaries」と呼ばれるフォルダがあり、その中で win32、win64、linux32、linux64 などの OS と bit 数を組み合わせたバイナリコードを保存するサブフォルダが存在します（図 5-4）。取り込み実行する側のツールにより差はありますが、通常ではこれらのツールは決められた OS、bit 数で無ければ実行できません。このため予め OS と bit 数を決めて FMU を作成する必要があります。なお binaries 以下のサブフォルダは複数持つことが認められているので、FMU の中に複数の実行可能な dll／so を保存することが可能です。

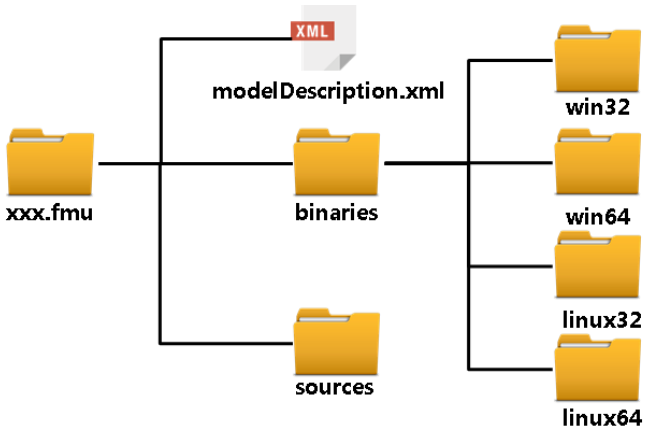


図 5-4　FMU の構造

5.3.3　隠蔽性とソースコードの利用

FMU にはソースコードを格納するフォルダ「sources」があり、この中にソースコードを格納することができます（図 5-4）。ソースコードの格納は任意ですので情報の隠蔽性を考える場合には、ソースコードを格納すべきではありません。一方で、OS 互換性のところで述べたように、実行可能な形式を binaries に保存して提供されないと実行できません。例えば ARM プロセッサ上で FMU を動作させたい場合、大半のツールで実行可能な FMU を作成できません。このような場合には、ソースコードを内部に持たせて作成し、取り込み側ツールでコンパイル、リンクを行います。

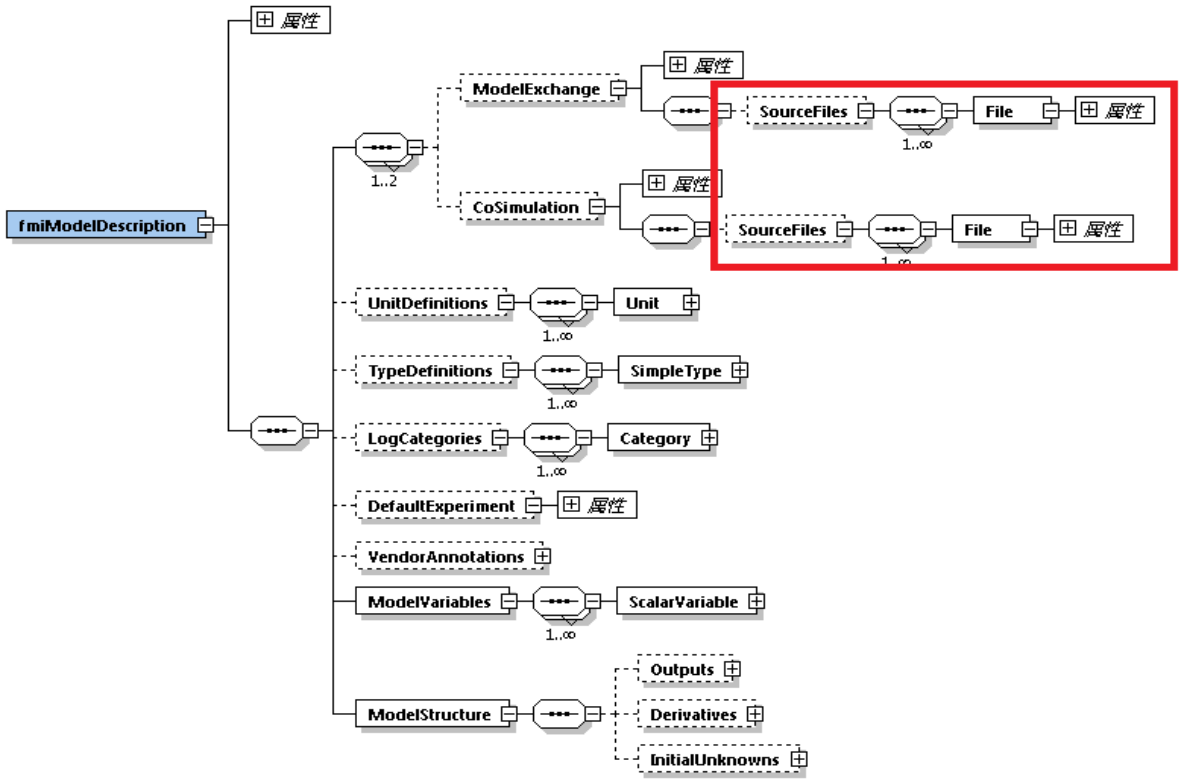


図 5-5　modelDescriptionFile の構造（対象部分のみ構成展開；赤色枠内）
（FMI 仕様書：FMI_for_ModelExchange_and_CoSimulation_v2.0 より引用）[7]

5.3.4　内部変数

FMU の生成ツールにより隠蔽の程度が異なりますが、入力・出力・パラメータ・開示変数しか外部から見ることができません。このため観測が必要な変数は意識的に外部から参照できるようにする必要があります。（図 2-5、2-10 および 5.2.1 参照）

5.4　各種ツールの注意点【FMI】

5.4.1　ライセンス

FMU を動作させるには、その FMU を生成したツールによっては、その生成環境が指定した実行用ライセンスを必要とするものがあります。ライセンスが必要な場合には、単に FMU を渡すだけでは使用できないので、生成側ツールの設定、取り込み側ツールのライセンス環境などを確認して下さい。

5.4.2　FMU の実行環境

FMU 内には、5.3.2 で述べたように取り込み側ツールの OS、bit 数に合ったものを生成する必要があります。また、CS の FMU としては、内部にソルバを持つものが多く利用されていますが、FMU 自身は通信の機能だけを持ち、シミュレーションにはソルバ自身を利用できる環境が必要となる場合もあります（2.4.1 参照）。この場合 5.4.1 に書いたライセンス環境だけではなく、ソルバ自身がインストールされた環境が必要です。

5.4.3　パラメータ変更

FMI の規定では FMU の中に実行前に変更可能な値（数値、論理値、文字列）をパラメータとして定義することができます。取り込み側ツールではその値を変更できないものもあります。取り込み側ツールで変更できない場合 FMU を再生成して渡さざるを得ないので、実際に使用する値を生成時に取り込み側と決めておく必要があります。なお FMU の中の xml ファイル中にデフォルト値が書かれていますが、この値を変更してもパラメータの値は反映できません。

5.4.4　名称

FMI では名称ルールが決まっています。FMU そのもの名称や、変数の名称（入出力、パラメータ）などで使用可能な文字列は Unicode の文字列とされています。

ツールに取り込む時に不要なエラーを発生させないためには、アルファベットで始まり、アルファベット、数字と“_”での組み合わせを使用することをお勧めします。

5.5　FMI とエラー

FMI を用いたモデルにおけるエラーも、FMI 特有のものと一般的なツール連携のものがあります。5.5.2 以降に各種エラーについて説明しますが、生成ツールで取り込むことができる場合、まず生成したツール自身で取り込みから実行まで実施し、問題ないことを確認するのが最初の確認事項だと考えます。

エラーメッセージは、取り込んだツールが出している場合と、FMU 自身が出している場合の二つの場合があります。どちらが出しているのかを確認できれば解決の糸口になります。FMU が出しているエラーは、その FMU を生成した際の元になるソースコードに記載されている可能性があります。

5.5.1　FMU の FMI 規格適合検証ツール

4.3.2 に記載しているように FMU が FMI に合わせて生成されているかを確認するツールとして Compliance Checker が存在します。

5.5.2　取り込みエラー【FMI】

FMI 特有の問題です。FMU を取り込むツールで FMU を取り込もうとする時に、エラーが発生する場合があります。この場合 5.5.1 の Compliance Checker にてチェックを行うことが推奨されます（6.1 参照）。5.4.4 で説明した名称の付け方によるエラーの場合も多く存在します。

5.5.3　初期値エラー【共通】

初期値エラーは 2 種類に分類できます。いずれも FMI 特有の問題ではありません

5.5.3.1　初期値不整合

2 つの FMU 同士、もしくは FMU と取り込みツール側のモデルの双方で同じ変数に対して初期値が規定されていて、その値が異なる場合のエラーです。対処方法は、どちらかの初期値の規定を解除することです。

5.5.3.2　初期値算出エラー

初期値を明示的に与えていない場合、シミュレーションを実行するツールの中で初期値推定を行うものがあります。その初期値推定ができない場合に出るエラーで、明示的に初期値を与えることで解決します。FMI では FMU 生成時に初期値を明示的に与えることができますが、前述の「初期値不整合」にならないように設定する必要があります。初期値をパラメータとして与えることができる FMU 生成ツールであれば、パラメータとして与えることが望ましいと考えます。

5.5.4　実行時エラー

5.5.4.1　初回シミュレーションエラー【FMI】

初期値算出エラーと似ていますが異なるエラーです。FMI の規定では初回シミュレーション時（例えば計算開始時刻を 0 とした場合は time=0 の時）に FMU に入力される値は 0「ゼロ」となります。この入力信号値を用いてシミュレーションしてしまった結果として、ゼロ割によるエラーが起こることが報告されています。

5.5.4.2　発散【共通】

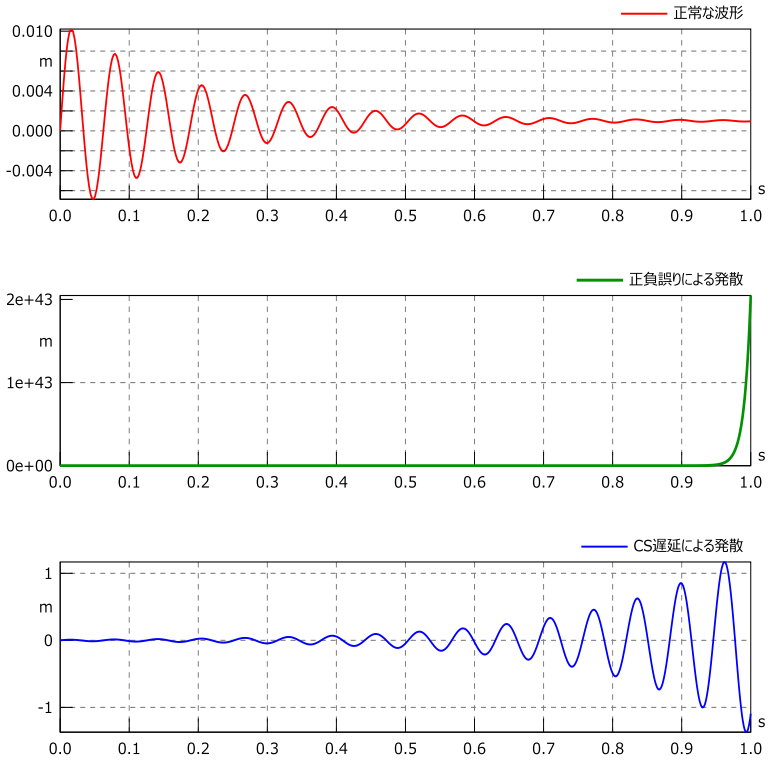
いずれも FMI 特有の問題ではありませんが次の 1) 2) は代表的なエラーです。

1)　シミュレーションの途中から次第に値が大きくなり、シミュレーションが異常終了する。

プラント同士の接続で頻繁に見受けるケースとして、物理量の符号が逆転している場合が考えられます。ネガティブフィードバックであるべきところが、ポジティブフィードバックになってしまっている結果です（図 5-6 中段）。5.1.3 に示したように、エネルギの方向との関係を整理し、生成側ツールで修正後再度 FMU を生成するか、読込側ツールで符号を反転させるかしなければなりません。

2)　シミュレーションの途中から信号が振動を起こしシミュレーションが異常終了する。

プラントモデルでの高い固有周波数が影響し、その固有周波数よりも低い周波数の信号授受（による遅延）が影響して不安定になっているケースが考えられます（図 5-6 下段）。CS の FMU では多くの場合 Communication Step Size を小さくすると回避できます。ME の FMU では生成環境のソルバで実行すると安定でも、取り込んだ実行環境のソルバでは安定性の違いによりエラーが発生することも多いので、実行ツールのソルバ環境を生成ツール側で理解して、できるだけ種類の近いソルバを用いて実行確認しておくことが望ましいと考えます（4.3.4 参照）。



上段：正常波形

中段：1）物理量の符号誤設定による発散

下段：2）信号授受間隔による発散

図 5-6　発散

5.5.4.3　　非収束【共通】

シミュレーションが先に進まず、計算が打ち切られる場合です。

可変ステップソルバを用いたツールで ME の FMU を使用している場合に発生することが多いエラーです。最小時間刻み幅を小さくするなどの対策を行います。多くの場合現象を不連続にするような事象（イベント）の発生が原因になっていますので、モデル自身に含まれるイベントを起こす部分を見つけ修正する必要があります。プラント同士の接続では、5.2.3 で述べたような「不連続性の高い部位の例」を参照して下さい。また、非因果ツール等で代数拘束条件（代数ループ）を与えた場合、同一時間内での代数方程式をソルバが解こうとするため、条件によっては解が得られずこのような現象が発生します（4.3.1 参照）。

第 6 章　チュートリアル

この章では、FMI を活用するための実践的な例題を用意しましたので、使用方法は順を追って説明していきます。6.1 では入手した FMU に問題が無いか確認するための Compliance Checker の使用方法について説明します。次に 6.2、6.3 では、これまでに当 WG で活動を進めてきたベンチマークモデルの 2 例について，概要、必要な設定、シミュレーション結果を掲載しました。[8][14]

6.1　Compliance Checker の使い方

4 章、5 章でも紹介している Compliance Checker の使用方法について説明します。

Compliance Checker は FMI ホームページよりダウンロードができます。

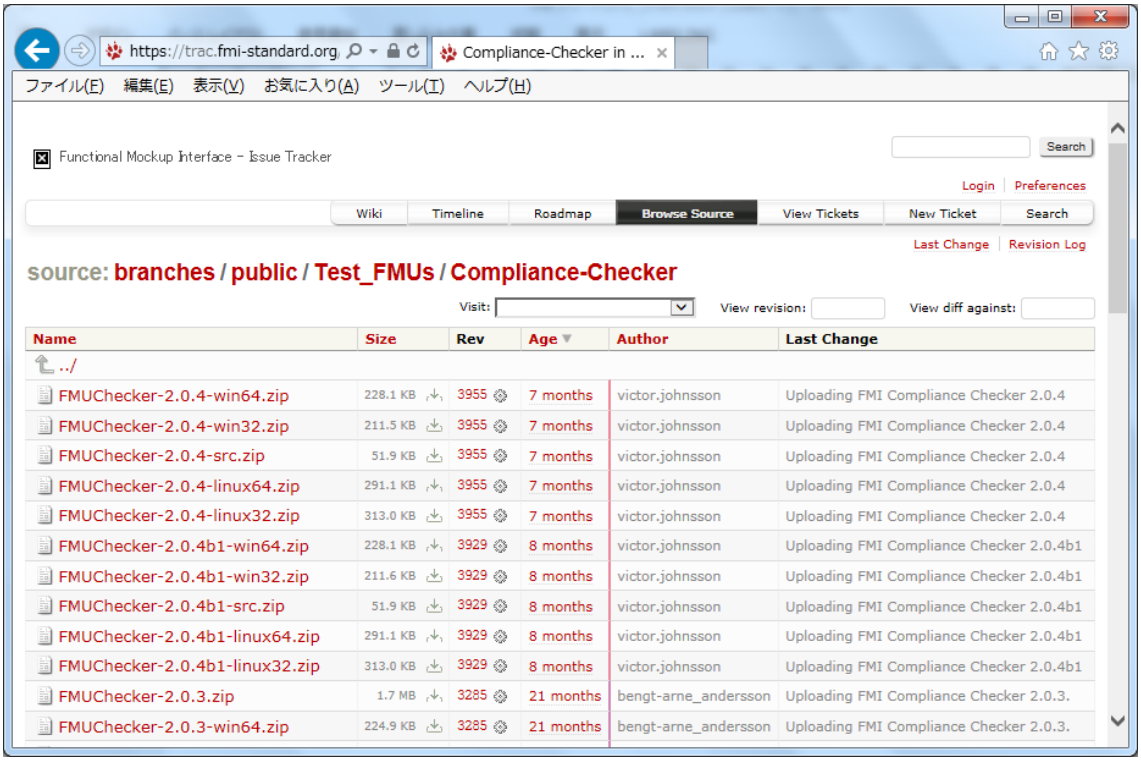


図 6-1　Compliance Checker の入手

図 6-1 はダウンロードページです。利用する OS 環境にあった最新版（”Age 列”を確認）をクリックして、ダウンロードして下さい。今回は Windows64bit 環境を例に、2018 年 6 月現在の最新 FMUChecker-2.0.4-win64.zip を使用します。

ダウンロードした ZIP ファイルを解凍し、任意のローカルドライブフォルダへ保存しましょう。

（以降の説明では任意フォルダを D:¥FMUChecker-2.0.4-win64 としています）

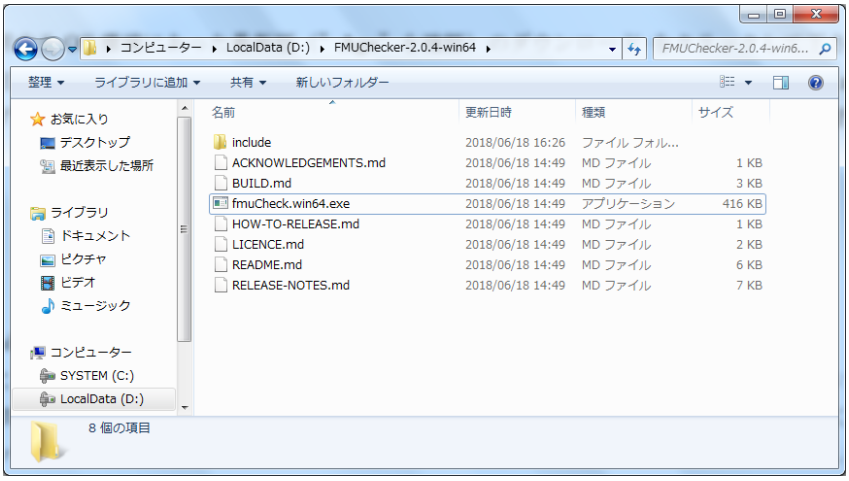


図 6-2　Compliance Checker の保存

次に Compliance Checker に掛けるモデルを準備します。

ここでは、3.1 で使用した FMU を例に warning や error を含んでいるか確認するためのステップを説明します。

なお、サンプルモデルは自動車技術会ホームページの自動車制御とモデル部門委員会ページよりダウンロードできます。

ファイル名：Sample_3p1.zip　：ZIP ファイルから FMU_J1_Case1_Dymola_ME_2.fmu を取り出して先ほどの任意フォルダと同じ階層にコピーします。

コマンドプロンプトを開いて先ほどの任意フォルダに移動して以下のコマンドを入力します。

fmuCheck.win64 -e error.log FMU_J1_Case1_Dymola_ME_2.fmu「Enter」キー

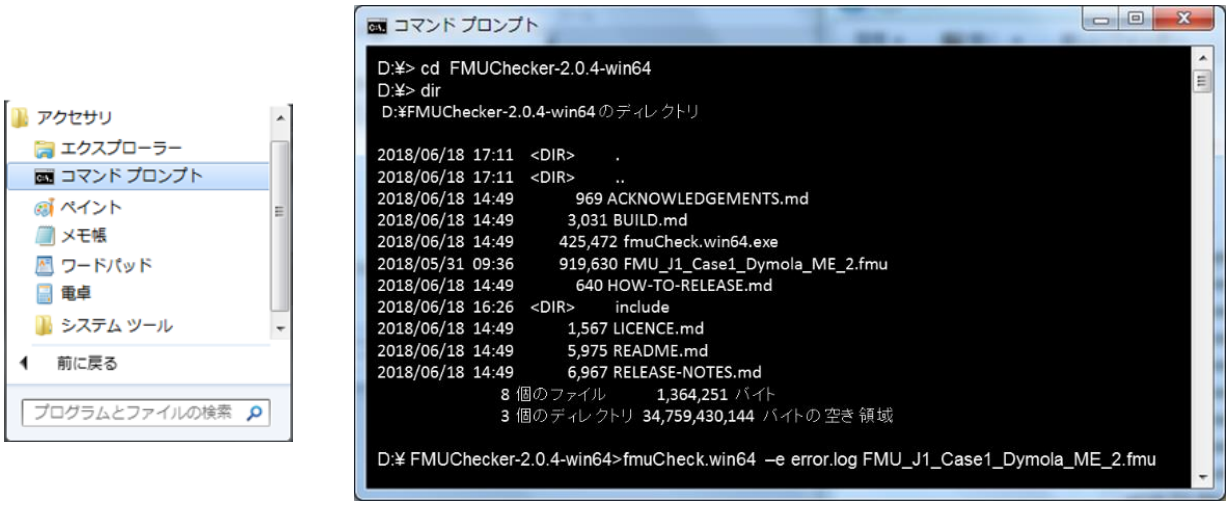


図 6-3　Compliance Checker でのコマンド入力画面

図 6-4 のように計算が流れて error.log が生成されます。（図 6-5）

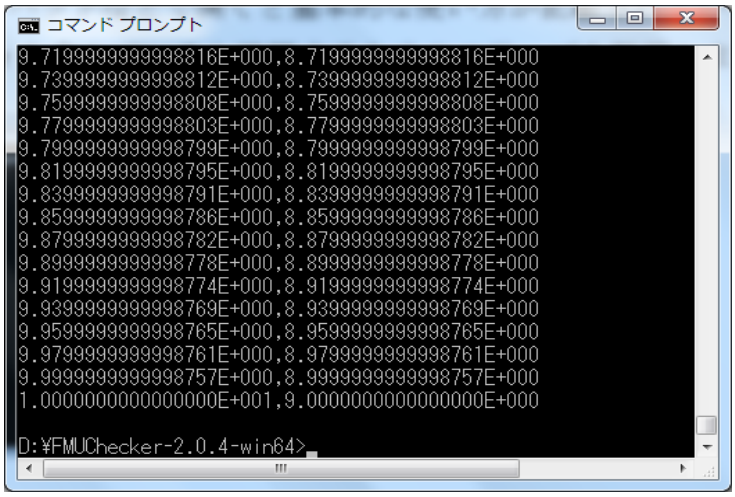


図 6-4　Compliance Checker による計算終了後の画面

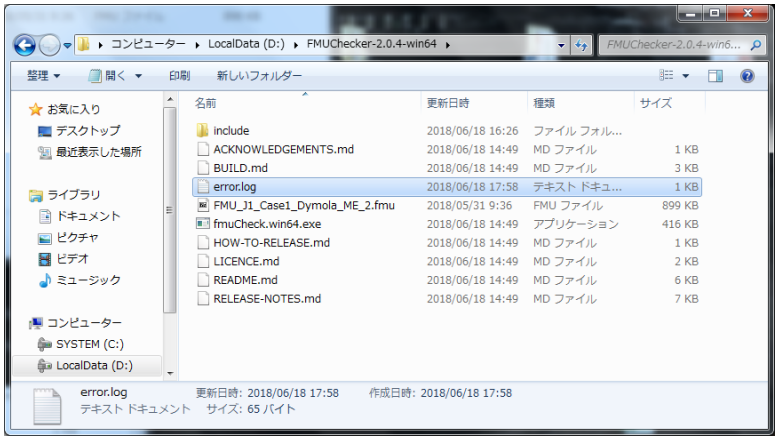
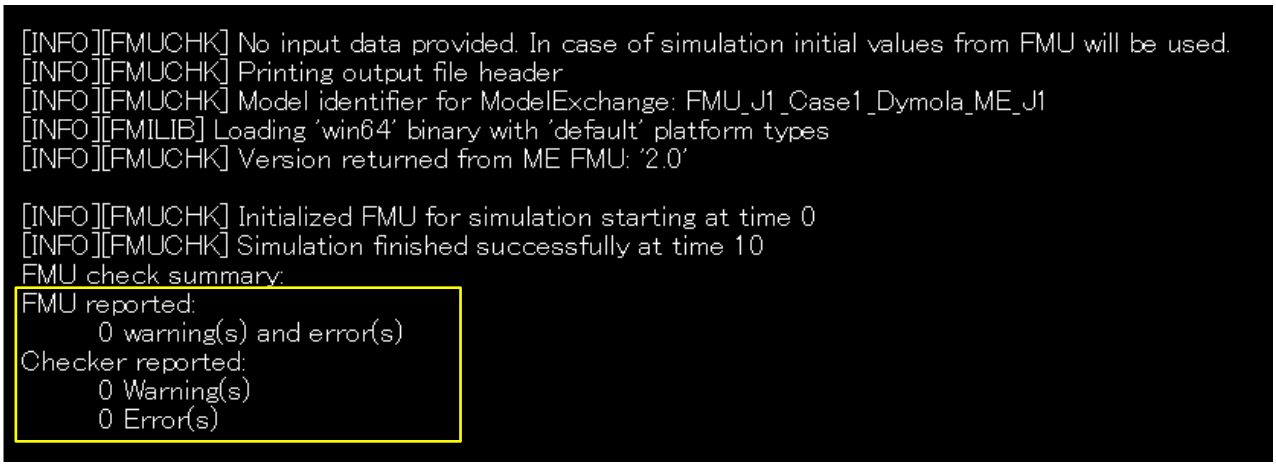


図 6-5　error.log の生成

error.log ファイルには以下の内容が保存されています。

```
[INFO][FMUCHK] FMI compliance checker 2.0.4 [FMILibrary: 2.0.3] build date: Nov  6 2017
[INFO][FMUCHK] Called with following options:
[INFO][FMUCHK] fmucheck.win64 -e ellor.log fmu_j1_case1_dymola_me_2.fmu
[INFO][FMUCHK] Will process FMU fmu_j1_case1_dymola_me_2.fmu
[INFO][FMILIB] XML specifies FMI standard version 2.0
[INFO][FMUCHK] Model name: FMU.J1.Case1.Dymola.ME.J1
[INFO][FMUCHK] Model GUID: {b718622a-62da-43d6-82bf-dfc618fe49ed}
[INFO][FMUCHK] Model version:
[INFO][FMUCHK] FMU kind: ModelExchange
[INFO][FMUCHK] The FMU contains:
1 constants
4 parameters
1 discrete variables¥
19 continuous variables
1 inputs
1 outputs
19 local variables
0 independent variables
0 calculated parameters
25 real variables
0 integer variables
0 enumeration variables
0 boolean variables
0 string variables
```



最下段 5 行を確認すると、この FMU にはエラーが無いことがわかります。

ご自身で入手した FMU を Compliance Checker に掛けて、エラーが表示された場合は、FMU 生成元へ問い合わせて正しく生成されたモデルを入手して下さい。

エラーが 0 と表示されてもシミュレーションツールによっては取り込みエラーとなることがあります。5.5.2 でもふれましたが、よくある原因として信号名に使用されている文字コードを取り込み側ツールが対応していないことがありますので使用するツールのベンダへ連絡し対応してもらって下さい。

Compliance Checker の詳細設定は同じ階層に置かれている README.md（図 6-5）に記載されているので、ワードパッド等で確認して下さい。

なお､現在の Compliance Checker バージョンでは FMI の細かな規約までチェックできていないため、エラーを見逃す可能性を否定できませんが、問題点の切り分けには大いに役立つことが期待できます。

6.2　例題 1：当 WG ベンチマークモデルの例

2017 年 10 月の自動車技術会 公開委員会「FMI (Functional Mockup Interface)によるモデル接続講習会」及び 2018 年 5 月の 2nd Japanese Modelica Conference で講演した当 WG のベンチマークモデル[8]を例題にチュートリアルを進めていきます。

6.2.1　サンプルモデルの説明

この車両モデルはドライバ、パワートレイン、ドライブライン、シャシ、ステアリングを 5 つの FMU で構成して Co-Simulation を行います。概略は図 6-6 を確認して下さい。

OS は Windows64bit 版、FMI2.0 のみに対応していますのでご注意ください。

サンプルモデルは自動車技術会ホームページの自動車制御とモデル部門委員会ページよりダウンロードできます。

ファイル名：Sample_6p2.zip　：ZIP ファイルには以下 7 個のファイルが格納されています。

Driver	: Driver_FMU_Export_Rev16.fmu.fmu
Power Train	: Powertrain_FMU_export_Rev16.fmu
Drive Line	: DS_Simulink.fmu
Chassis	: FMI_20_Blackbox_Euler_Test_FlatPad.fmu
Steering	: SteeringMechanismRackPinionInMetric.fmu
サブシステム I/F 定義書	: Sample_6p2_サブシステム I/F 定義書.xlsx
シミュレーション結果	: Sample_6p2_Result.txt

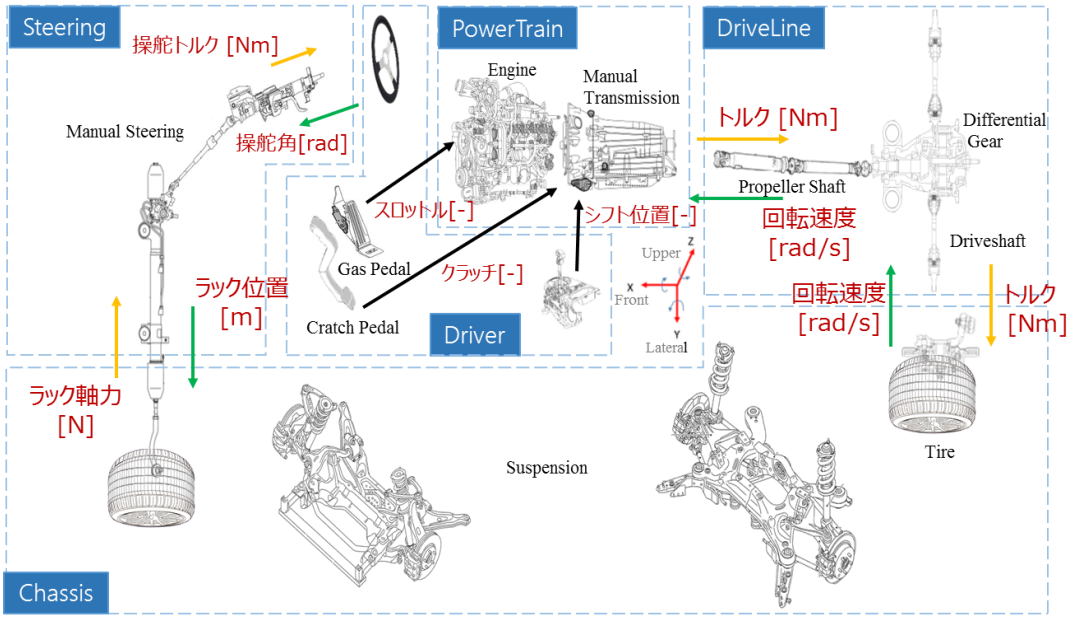


図 6-6　ベンチマークモデルの概要

まずは、使用するシミュレーションツールへ 5 つの FMU を取り込みます。取り込み方法は、メニュー画面で指定する場合や Explorer からドラッグ＆ドロップなどツールによって異なりますので、マニュアルを参照して下さい。なお、各 FMU の内容について知りたい場合は、ZIP ファイルに同梱したサブシステム I/F 定義書（Sample_6p2_サブシステム I/F 定義書.xlsx）に概要を記載しています。

6.2.2　モデルの接続

次に、各 FMU を接続します。表 6-1 を参考に接続して下さい。緑色のハッチングはモニタ用出力のため接続には使用しません。図 6-7 はモデル接続例です。各 FMU の左側端子は入力、右側端子は出力になっています。使用するツールにより接続時の体裁は異なるためご注意ください。

表 6-1　入出力一覧表

FMU番号	端子番号		端子番号	入力端子			FMU番号	FMU名称	出力端子			端子番号
							1	driver	Clutch_Action@expseu_	[-]	クラッチ	1
									Engine_load@expseu_	[-]	スロットル開度	2
									Gear_position@expseu_	[-]	シフト位置	3
									Steerin_Angle@expseu_	[rad]	操舵角	4
3	1	⇒	1	Npropshaft@expseu_	[rad/s]	Prop/Shaft回転速度	2	Power Train	Neng@expseu_	[rad/s]	エンジン回転速度	1
1	3	⇒	2	Gear_position@expseu_	[-]	シフト位置			Nclutch@expseu_	[rad/s]	クラッチ側回転速度	2
1	2	⇒	3	Engine_load@expseu_	[-]	スロットル開度			Tpropshaft@expseu_	[Nm]	Prop/Shaftトルク	3
1	1	⇒	4	Clutch_Action@expseu_	[-]	クラッチ						
2	3	⇒	1	Tq_TM	[Nm]	Prop/Shaftトルク	3	Drive Line	N_Propshaft	[rad/s]	Prop/Shaft回転速度	1
4	7	⇒	2	N_RL	[rad/s]	後左輪D/Shaft回転速度			Tq_RL	[Nm]	後左輪D/Shaftトルク	2
4	5	⇒	3	N_RR	[rad/s]	後右輪D/Shaft回転速度			Tq_RR	[Nm]	後右輪D/Shaftトルク	3
5	2	⇒	1	steeringTranslation	[m]	ラック位置	4	Chassis	position_x	[m]	車両重心位置X方向	1
固定値[0]		⇒	2	FrontRightWheelTau	[N]	前右輪D/Shaftトルク			position_y	[m]	車両重心位置Y方向	2
3	3	⇒	3	RearRightWheelTau	[N]	後右輪D/Shaftトルク			position_z	[m]	車両重心位置Z方向	3
固定値[0]		⇒	4	FrontLeftWheelTau	[N]	前右輪D/Shaftトルク			FrontRightWheel_w	[rad/s]	前右輪回転速度	4
3	2	⇒	5	RearLeftWheelTau	[N]	後右輪D/Shaftトルク			RearRightWheel_w	[rad/s]	後右輪回転速度	5
									FrontLeftWheel_w	[rad/s]	前左輪回転速度	6
									RearLeftWheel_w	[rad/s]	後左輪回転速度	7
									steering_force	[N]	ラック軸力	8
1	4	⇒	1	steerAngle	[rad]	操舵角	5	Steering	steerTorque	[Nm]	操舵トルク	1
4	8	⇒	2	rackForce	[N]	ラック軸力			rackPosition	[m]	ラック位置	2

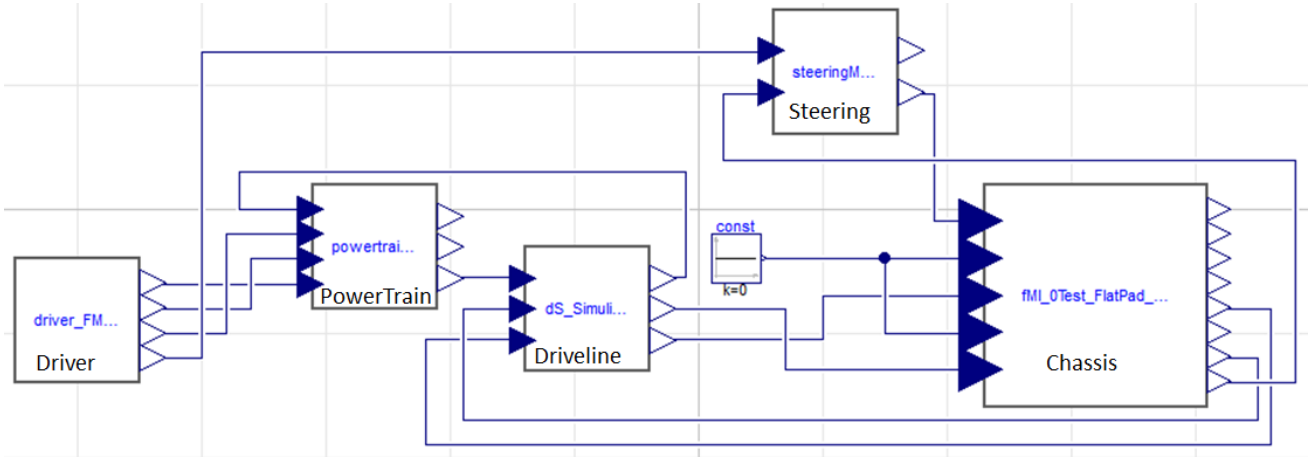


図 6-7　モデル接続例

6.2.3　ドライバモデルの説明

図 6-8 に示したように Open loop の単純な入力操作としています。マニュアルトランスミッション仕様のため、クラッチ操作を必要とし、停止から 5 速ギヤまで順次変速していきます。

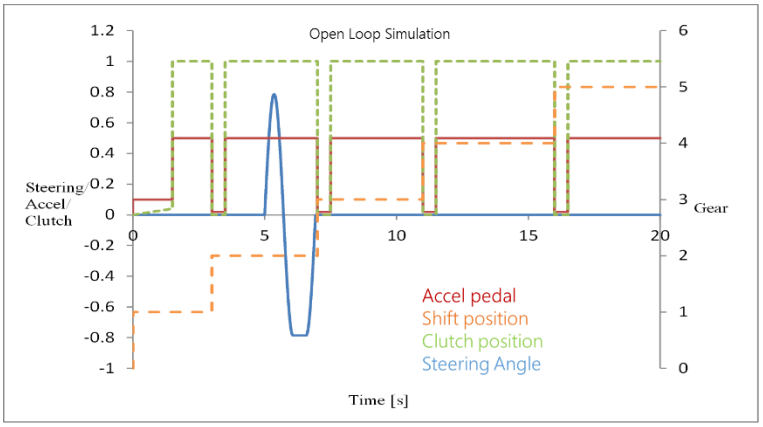


図 6-8　ドライバモデルの動作

6.2.4　FMU パラメータの設定

デフォルトのパラメータでは、正方向（左転舵）のステアリング操作に対してタイヤが右に切れてしまうため、設定を変更する必要があります。図 6-9 はステアリングモデルの概要です。ドライバモデルからの入力”steerAngle”に対して、”rackPosition”の移動方向を反転させるため、赤枠で示した”gs1”ブロックのパラメータを変更します。

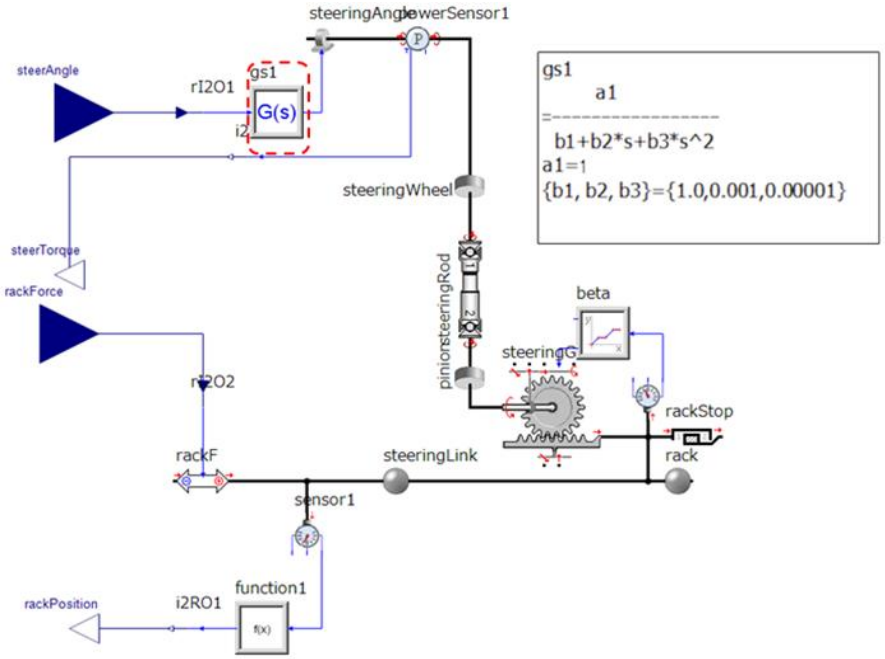


図 6-9　ステアリングモデルの概要

SimulationX の例では、図 6-10 のように Parameters のタブで設定します。

- 2 次伝達関数の分子（gs1.A[1]）：現在の ”1 [-]” から ”－1[-]” へ変更

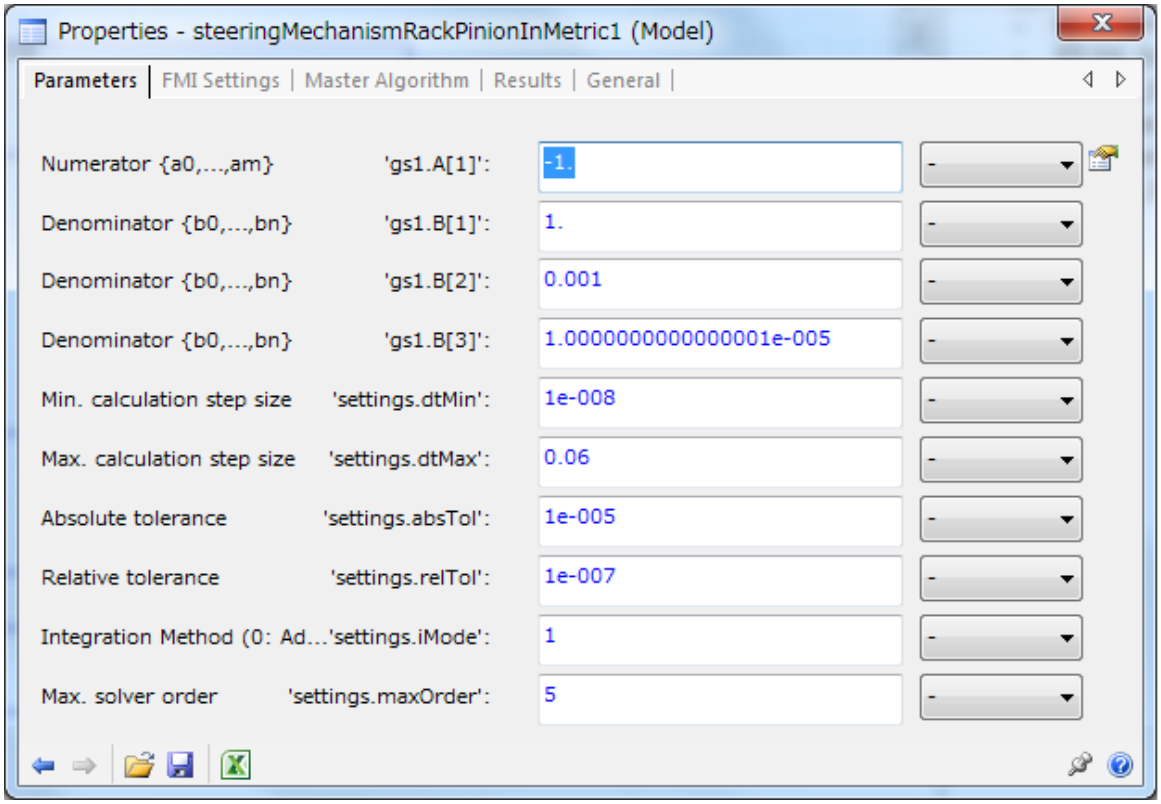


図 6-10　ステアリングのパラメータ設定（SimulationX での例）

6.2.5　マスタツールのシミュレーション設定

ドライバモデルで一連の操作が完了する時間の 20 秒をシミュレーション時間として設定します。
ソルバは固定ステップの”ODE1”相当とします。
最少ステップサイズは、5 つの FMU で、最も細かい計算間隔に合わせて”5e-5 [s]”とします。

Start Time	: 0 [s]
Stop Time	: 20 [s]
Solver, Step Sizes and Tolerances	: Fixed step Solver（固定ステップ）
Integration method	: Euler Forward　（ODE1 相当）
Min. Calculation Step Size	: 0.00005　（または 5e-5）[s]
Min. Output Step Size	: 0.01 [s]　・・・出力結果のサンプル時間なので任意

SimulationX の例では SimulationControl で設定します。

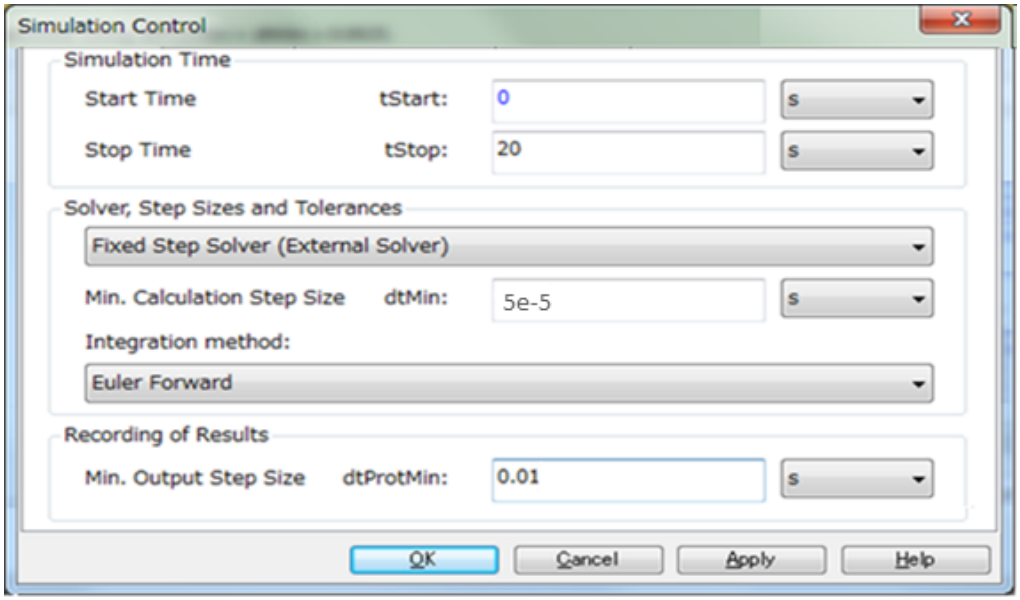


図 6-11　シミュレーション設定（SimulationX での例）

この例題において、マスタツール上は FMU のみで構成される Co-Simulation のため、ソルバの違いで大きな差は生じませんが、一般的な使われ方ではマスタツール上のモデルを計算し、さらに取り込んだ FMU との間で信号を受け渡すので、ソルバ、ステップサイズは慎重に設定する必要があります。

6.2.6　Communication Step Size の設定

Co-Simulation では FMU 毎に入力信号を受け取る時間間隔設定が必要となります。今回のモデルでは表 6-2 に従って全ての FMU に設定を行います。

表 6-2　各 FMU の Communication Step Size 設定

Setting	Driver	Power Train	Drive Line	Chassis	Steering
Communication Step Size	5.00E-05	5.00E-05	5.00E-05	1.00E-03	1.00E-03

SimulationX の例では Master Algorithm のタブで設定します。各 FMU を同様に設定して下さい。

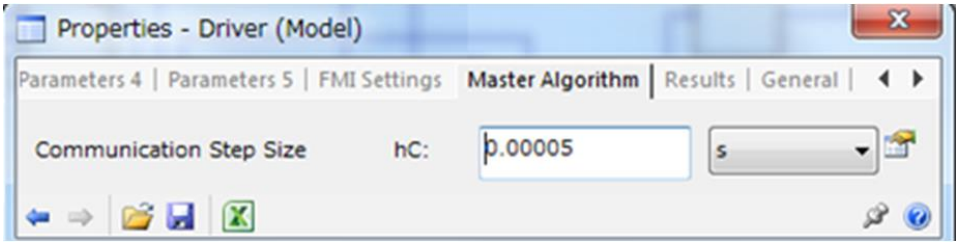


図 6-12　Communication Step Size の設定（SimulationX での例）

なお、Communication Step Size はツールによって名称が異なるため、代表例を紹介します。

Amesim　　: Co-simulation Step Size
Dymola　　: fmi_Communication Step Size
MapleSim　　: MapleSim Sync Step
Simplorer　　: TS Simplorer
SimulationX　　: Communication Step Size hC

6.2.7　シミュレーション結果：

各 FMU からの出力信号をいくつか掲載しました。同じような結果となったでしょうか？

詳細は Sample_6p2_Result.txt にシミュレーション結果を保存しているため参考にして下さい。

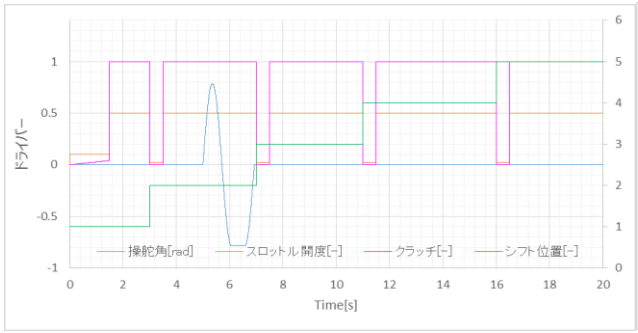


図 6-13a　FMU1（出力）Driver

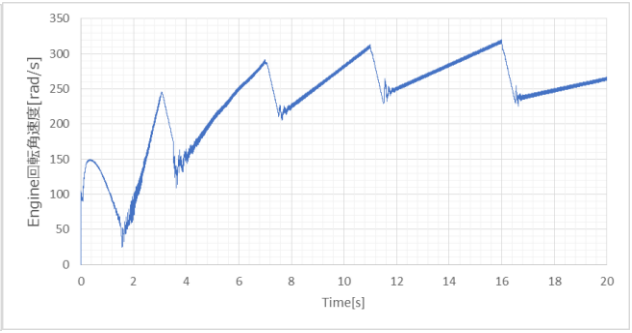


図 6-13b　FMU2（出力）エンジン回転速度

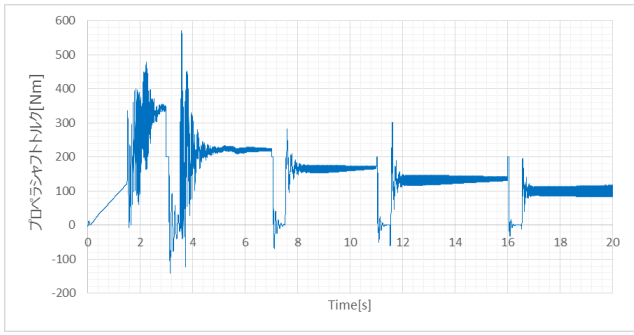


図 6-13c　FMU2（出力）Prop/Shaft トルク

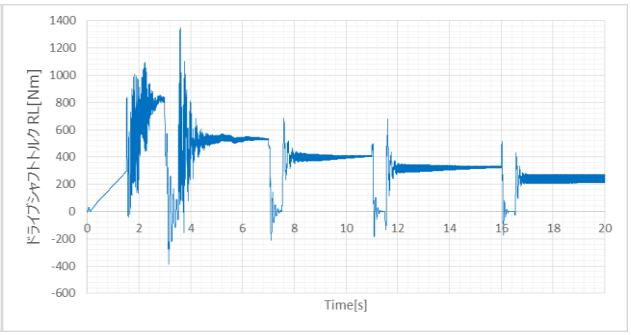


図 6-13d　FMU3（出力）後左軸 D/Shaft トルク

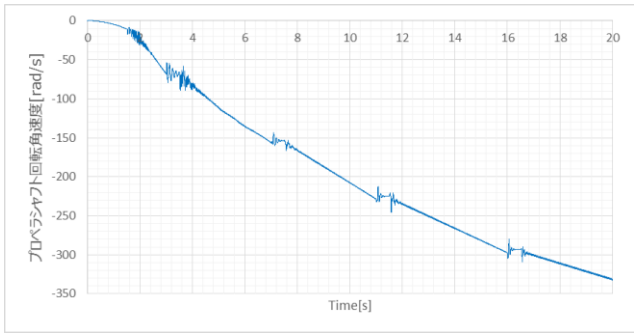


図 6-13e　FMU3（出力）Prop/Shaft 回転速度

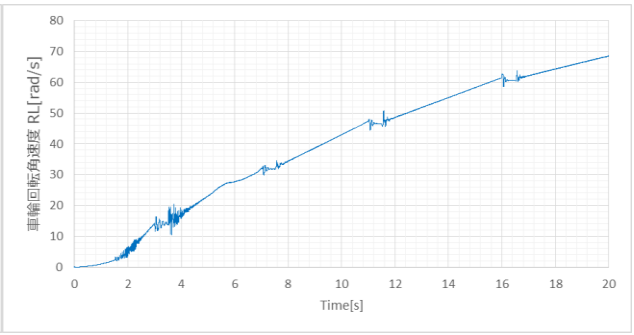


図 6-13f　FMU4（出力）後左軸 D/Shaft 回転速度

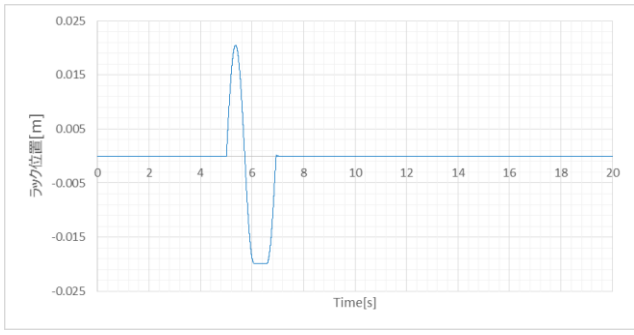


図 6-13g　FMU4（出力）ラック位置

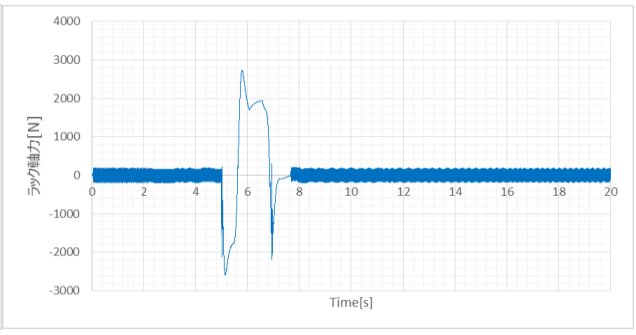


図 6-13h　FMU5（出力）ラック軸力

6.3　例題 2：経産省ガイドラインに準拠したベンチマークモデルの例

第 4 章、5 章でも述べたとおり、2017 年 3 月に経済産業省の自動車産業におけるモデル利用のあり方に関する研究会より、ガイドラインが示され準拠モデル Ver.1.0（準拠モデルは Simulink ベースのモデル）が同時に公開されました。詳細は経済産業省ホームページで確認して下さい。

6.3.1　サンプルモデルの説明

ここからは、2018 年 5 月の自動車技術会春季大会で講演した当 WG のベンチマークモデルを例題にチュートリアルを進めていきます。[13]

経産省ガイドライン準拠モデルをベースに 7 つの FMU を作成しました。

Driver model（Split1）、Vehicle model については 6 つ（Split2～Split7）に分割しています。

サンプルモデルは自動車技術会ホームページの自動車制御とモデル部門委員会ページよりダウンロードできます。FMI2.0 Windows 64bit CS のみに対応していますのでご注意ください。

ファイル名：Sample_6p3.zip　　：ZIP ファイルには以下 9 個のファイルが格納されています。
FMU　：Driver.fmu，ENG_CNT.fmu，TM_CNT.fmu，PWT_PNT.fmu，CHA_PNT.fmu，
　　ALT_CNT.fmu，ELEC_PNT.fmu
JC08 モードでの目標車速の時系列データ　：jc08.csv
シミュレーション結果　　：Sample_6p3.txt

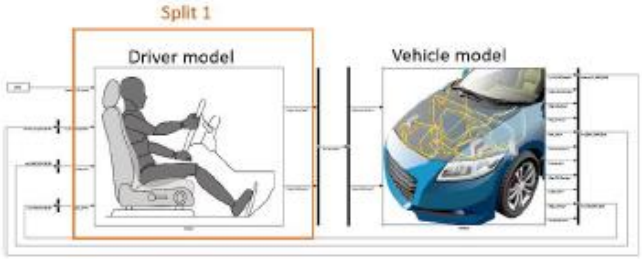


図 6-14　経業省ガイドライン準拠モデル 最上位階層の構成

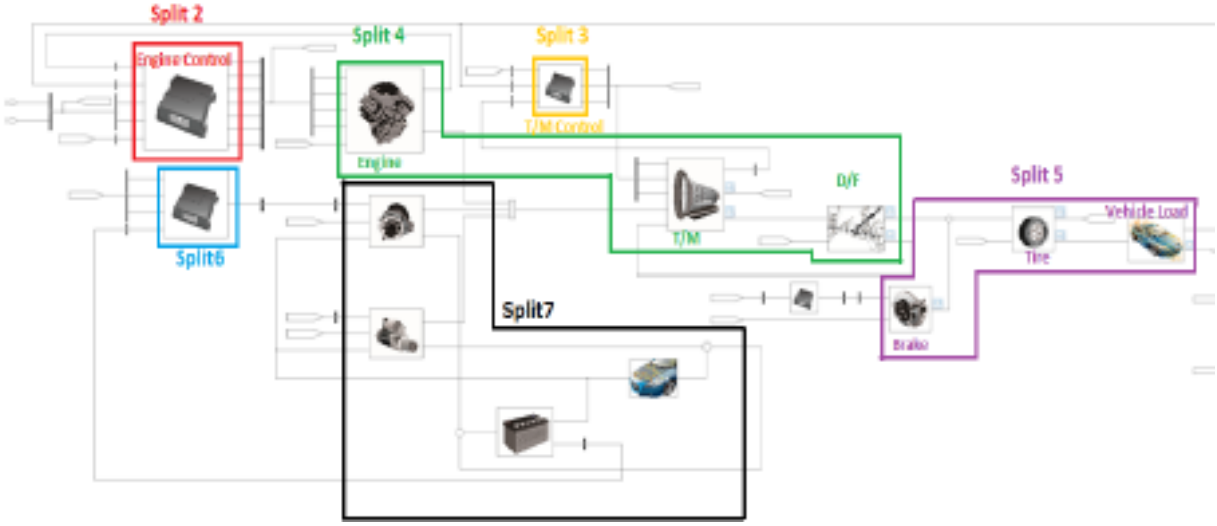


図 6-15　経産省ガイドライン準拠モデル Vehicle model の構成

まずは、使用するシミュレーションツールへ 7 つの FMU を取り込みます。取り込み方法は、メニュー画面で指定する場合や Explorer からドラッグ＆ドロップなどツールによって異なりますので、マニュアルを参照して下さい。

6.3.2　モデルの接続

次に、表 6-3 を参考に各 FMU 間を接続します。黄色ハッチングは目標車速、緑色はモニタ用出力です。図 6-16 はモデル接続例です。各 FMU の左側端子は入力、右側端子は出力になっています。モニタ用端子については接続に使用しません。使用するツールにより接続時の体裁は異なるためご注意ください。

表 6-3 各 FMU の入出力信号

FMU Number	出力端子			入力端子		FMU Number	FMU Name	出力端子	
3	1	ratio_CVT	→	1	ratio_CVT	1	Driver	open_brake_per	1
2	5	n_eng_rpm_sig	→	2	n_eng_rpm_sig			open_accel_per	2
JC08 km/h			→	3	target_v_VL_kmph				
5	2	V_PNT_kmph	→	4	v_VL_PNT_kmph				
3	2	flag_lockup	→	1	flag_Lockup	2	ENG_CNT	open_throttle_per	1
4	2	n_eng_rpm	→	2	n_eng_rpm			flag_fuelcut	2
1	1	open_brake_per	→	3	open_brake_per			flag_IdelStop	3
1	2	open_accel_per	→	4	open_accel_par			timing_ignition	4
5	2	V_PNT_kmph	→	5	v_VL_PNT_kmph			n_eng_rpm_sig	5
								flag_ON_Starter	6
4	1	n_TM_PNT_rpm	→	1	n_TM_PNT_rpm	3	TM_CNT	ratio_CVT	1
5	2	V_PNT_kmph	→	2	v_VL_PNT_kmph			flag_Lockup	2
2	1	open_throttle_per	→	3	open_throttle_per			omg_Slip_rpm	3
5	1	w_TR_PNT_radps	→	1	w_TR_PNT_radps	4	PWT_PNT	n_TM_PNT_rpm	1
3	1	ratio_CVT	→	2	ratioCVT			n_eng_rpm	2
3	2	flag_Lockup	→	3	flag_Lockup			trq_DF_PNT_Nm	3
3	3	omg_slip_rpm	→	4	omg_Slip_rpm			tScope_Fuel	4
2	1	open_throttle_per	→	5	open_throttle_per			tScope_Fuelratio	5
2	2	flag_fuelcut	→	6	flag_fuelcut			tScope_CVTLoss	6
2	3	flag_IdleStop	→	7	flag_IdleStop			tScope_trq_Flywheel2	7
2	4	timing_ignition	→	8	timing_ignition			w_ENG_PNT_radps	8
7	2	trq_ALT_PNT_Nm	→	9	trq_ALT_PNT_Nm				
7	3	trq_ST_PNT_Nm	→	10	trq_ST_PNT_Nm				
1	1	open_brake_per	→	1	open_brake_per	5	CHA_PNT	w_TR_PNT_radps	1
4	3	trq_DF_PNT_Nm	→	2	trq_DF_PNT_Nm			v_VL_PNT_kmph	2
								tScope_V_VL_PNT_mps	3
7	1	SOC_BT_PNT_Lo_PCT	→	1	SOC_BT_PNT_Lo_PCT	6	ALT_CNT	target_volt_ALT_V	1
2	2	flag_fuelcut	→	2	flag_fuelcut				
2	3	flag_IdelStop	→	3	flag_IdleStop				
2	5	n_eng_rpm_sig	→	4	n_eng_rpm_sig				
6	1	target_volt_ALT_V	→	1	target_volt_ALT_V	7	ELEC_PNT	SOC_BT_PNT_Lo_PCT	1
2	6	flag_ON_Starter	→	2	flag_ON_Starter			trq_ALT_PNT_Nm	2
4	8	w_ENG_PNT_radps	→	3	w_ENG_PNT_radps			trq_ST_PNT_Nm	3
								tScope_pulley1Loss	4
								tScope_SOC_Batt	5
								tScope_V_ALT	6

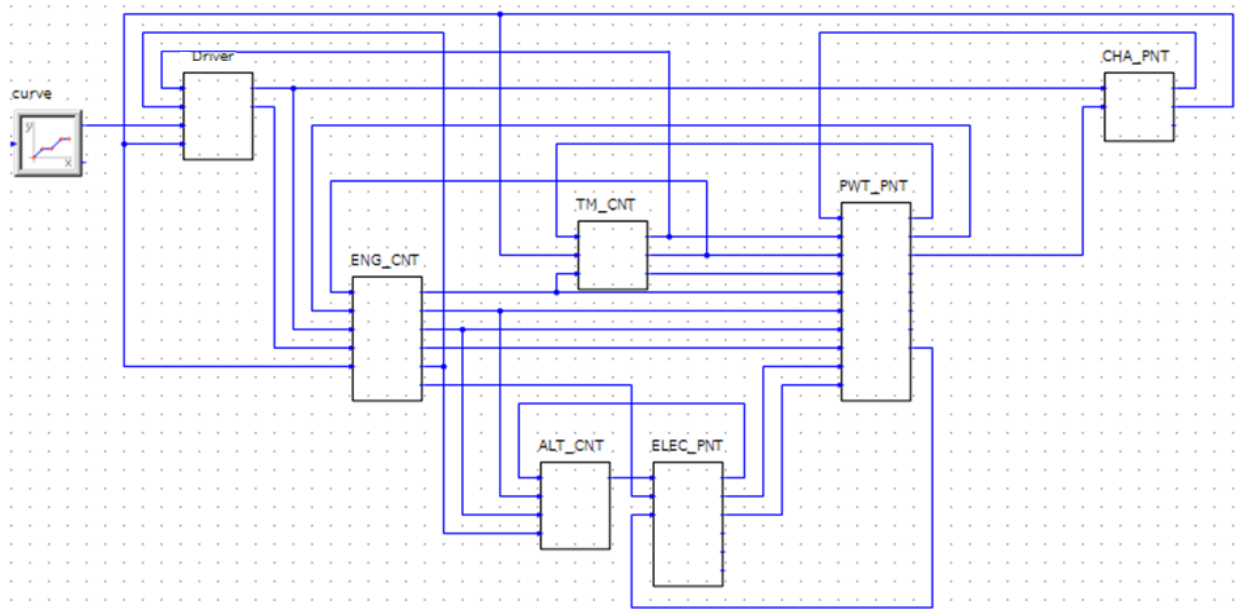


図 6-16　モデル接続例（SimulationX での例）

6.3.3　ドライバモデルの説明

ドライバモデル（FMU1:Driver）は、目標車速（表 6-3 黄色ハッチング部）となるようアクセルとブレーキを Closed Loop で操作します。今回は、JC08 モードで走行させるため jc08.csv の時系列データを入力します（図 6-17）。図 6-16 の接続例では curve へ設定します。

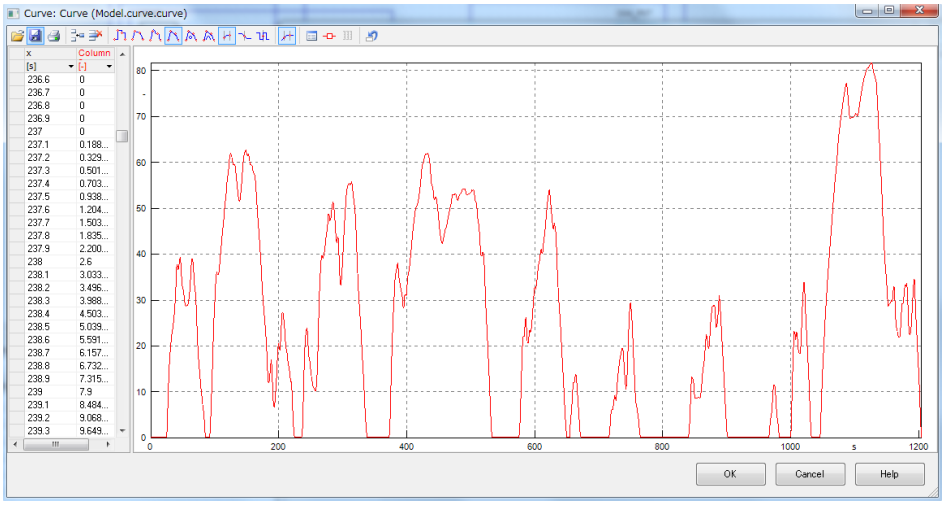


図 6-17　目標車速

6.3.4　マスタツールのシミュレーション設定

JC08 モードの所要時間である 1204 秒をシミュレーション時間として設定します。

ソルバは固定ステップの”ODE1”相当とします。

最少ステップサイズは、使用する各 FMU の最少計算間隔に合わせて”2.5e-3 [s]”とします。

Start Time	: 0 [s]
Stop Time	: 1204 [s]
Solver, Step Sizes and Tolerances	: Fixed step Solver（固定ステップ）
Integration method	: Euler Forward　（ODE1 相当）
Min. Calculation Step Size	: 0.0025　（または 2.5e-3）[s]
Min. Output Step Size	: 0.01 [s]　・・・出力結果のサンプル時間なので任意

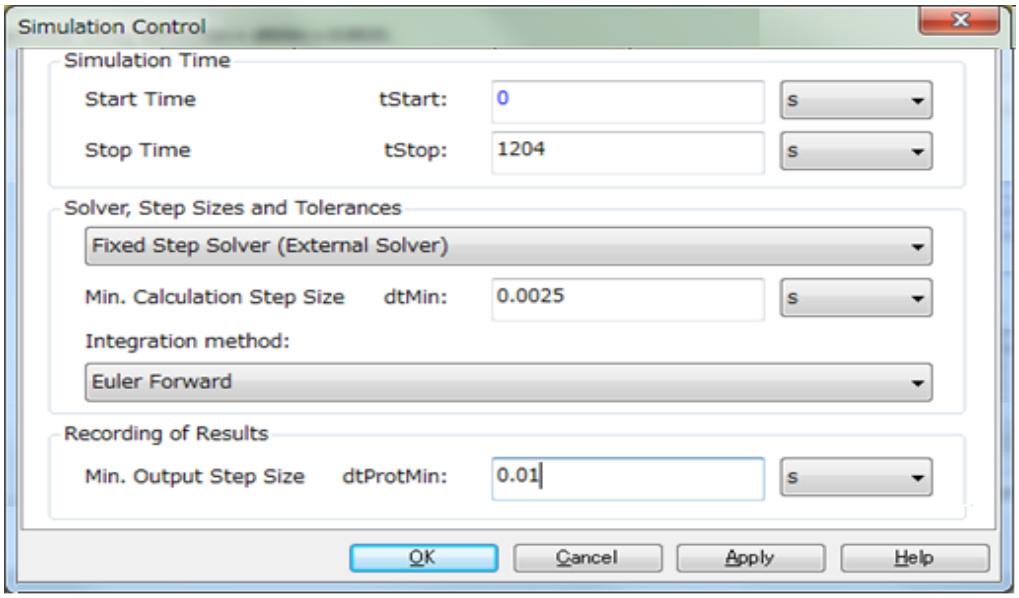


図 6-18　シミュレーション設定（SimulationX での例）

6.3.5　Communication Step Size の設定

Co-Simulation では FMU 毎に入力信号を受け取る時間間隔設定が必要となります。

今回のモデルでは全ての FMU を 0.0025（または 2.5e-3）[s]で設定します。

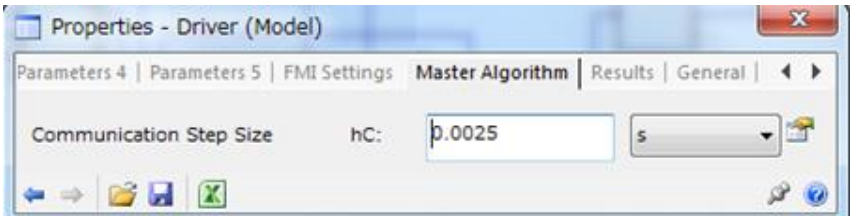


図 6-19　Communication Step Size の設定（SimulationX での例）

6.3.6　シミュレーション結果

FMU からの出力信号をいくつか掲載しました。

詳細は Sample_6p3.txt にシミュレーション結果を保存しているため参考にして下さい。

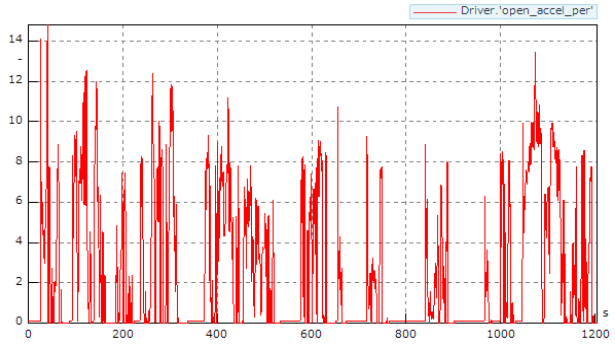


図 6-20a　FMU1　open_accel_per

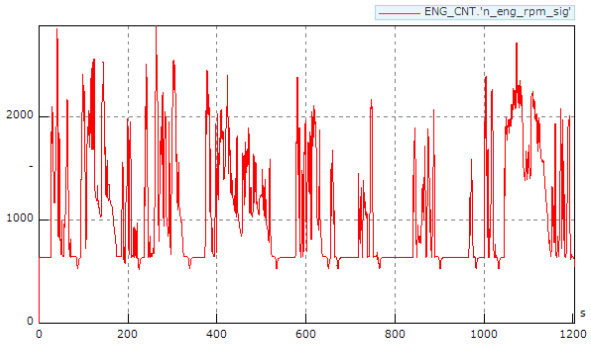


図 6-20b　FMU2　n_eng_rpm_sig

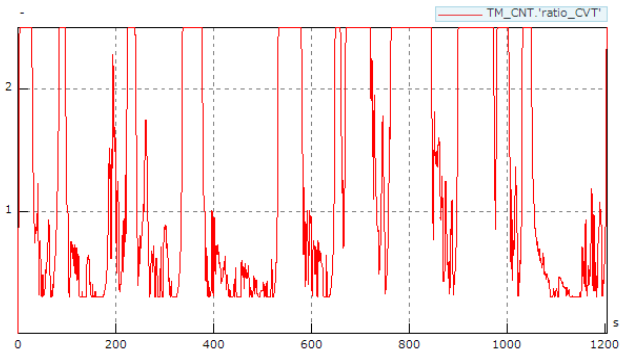


図 6-20c　FMU3　ratio_CVT

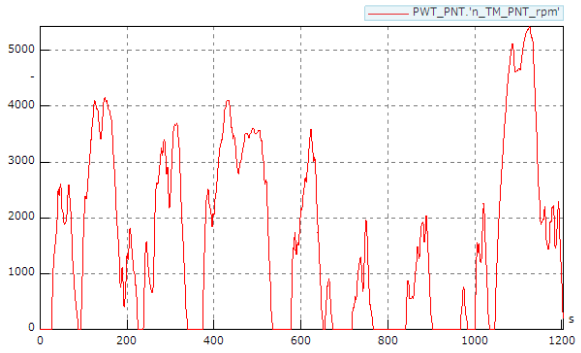


図 6-20d　FMU4　n_TM_PNT_rpm

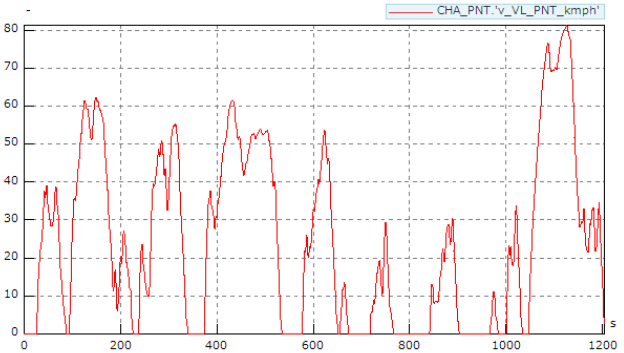


図 6-20e　FMU5　v_VL_PNT_kmph

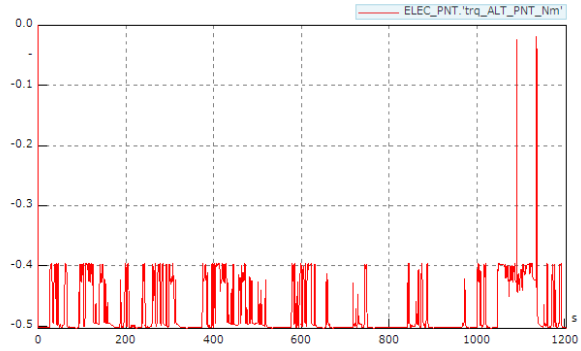


図 6-20f　FMU7　trq_ALT_PNT_Nm

第 7 章　引用文献

1. 平野 豊：FMI によるモデル流通を目指して，自動車技術会振動騒音部門委員会 No.10-17 シンポジウム,(2017)
http://www.jsae.or.jp/sympo/2017/docu/No_10-17.pdf

2. 経済産業省 自動車新時代戦略会議 ： 自動車新時代戦略会議（第 1 回）資料, (2018)
http://www.meti.go.jp/shingikai/mono_info_service/jidosha_shinjidai/pdf/001_01_00.pdf

3. Blochwitz Torsten, Martin Otter, et al.：The Functional Mockup Interface for Tool independent Exchange of Simulation Models, Preprint of the 8th International Modelica Conference,(2011)
https://trac.fmi-standard.org/export/700/branches/public/docs/Modelica2011/The_Functional_Mockup_Interface.pdf

4. 自動車技術会 国際標準記述によるモデル開発･流通検討委員会 モデル接続技術検討 WG： 非因果モデリングツールを用いた FMI モデル接続ガイドライン Ver.1.0, (2015)
http://www.jsae.or.jp/tops/topics/1241/1241-1A.pdf

5. Blochwitz Torsten, Martin Otter, et al.：The Functional Mockup Interface for Tool independent Exchange of Simulation Models, Proceedings of the 8th International Modelica Conference,(2011)
https://www.modelica.org/events/modelica2011/Proceedings/pages/papers/05_1_ID_173_a_fv.pdf

6. A. Junghanns, T. Blochwitz：10_Years_of_FMI Where are we now? Where we do go? , Keynote speech of the 2nd Japanese Modelica Conference, (2018)
https://www.modelica.org/events/modelica2018japan/presentation/10_Years_of_FMI.pdf

7. Modelica Association：Functional Mock-up Interface for Model Exchange and Co-Simulation Document version 2.0,(2014)
https://svn.modelica.org/fmi/branches/public/specifications/v2.0/FMI_for_ModelExchange_and_CoSimulation_v2.0.pdf

8. Yutaka Hirano, Junichi Ichihara, et al.：
Toward the actual using FMI in practical use cases in Japanese automotive industry [p195-203], Program of the 2nd Japanese Modelica Conference, (2018)
https://www.modelica.org/events/modelica2018japan/conference-proceedings/modelica-final-proceedings-2018-Japan.pdf

9. 日経デジタルヘルスホームページ ：Modelica と FMI―構想設計段階で役立つ CAE 規格（2013)
https://tech.nikkeibp.co.jp/dm/article/COLUMN/20131107/314661/

10. FMI ホームページ：

https://fmi-standard.org/

https://fmi-standard.org/tools/

https://trac.fmi-standard.org/browser/branches/public/Test_FMUs/Compliance-Checker

11. 経済産業省　自動車産業におけるモデル利用のあり方に関する研究会：
自動車開発におけるプラントモデル I/F ガイドライン（2017）

http://www.meti.go.jp/press/2016/03/20170331010/20170331010.html

http://www.meti.go.jp/press/2016/03/20170331010/20170331010-1.pdf

12. 緒方 洋介、村上 晋太郎、他：FMI, Model Exchange, Co-simulation におけるシミュレーション安定性に関する考察、自動車技術会 2018 年春季大会 S4-4,(2018)

https://www.gakkai-web.net/gakkai/jsae/s/pro2018/session_ab/4.html

13. 市原 純一、斉藤 春樹、他：複数モデリングツールによる FMI を用いた Co-Simulation に関するモデル接続活動紹介（第 2 報）、自動車技術会 2018 年春季大会 S4-2,(2018)

https://www.gakkai-web.net/gakkai/jsae/s/pro2018/session_ab/4.html

14. 平野 豊、関末 崇行：FMI によるモデル流通を目指して，自動車技術会 2018 年春季大会 S4-1, (2018)

https://www.gakkai-web.net/gakkai/jsae/s/pro2018/session_ab/4.html

第 8 章　索引

Appendix

modelDescriptopnScheme　(fmi2ModelDescription.xsd)