

信頼性システム



モデリング言語「SafeML」

ユーザーズガイド Version 2.0



SafeML2.0 を活用し、リスクアセスメントを行う方法を示したユーザーズガイド

2022 年 6 月

ロボット革命・産業 IoT イニシアティブ協議会

ロボットイノベーション WG (WG3)

ソフトウェアアーキテクチャ調査検討委員会

発行者 ロボット革命・産業 IoT イニシアティブ協議会
〒105-0011 東京都港区芝公園 3-5-8
機械振興会館 507 号室 日本機械工業連合会内
TEL 03-3434-6571
E-mail office@jmfri.gr.jp
URL <https://www.jmfri.gr.jp/>

Copyright © 2022 ロボット革命・産業 IoT イニシアティブ協議会 All Rights Reserved.

本文書は、著作権法および国際条約により保護されています。個人または会社（または会社に準ずるもの）内部での使用を目的として、本文書をダウンロード、印刷、または電子的に閲覧することができます。本資料の内容の全部又は一部については、私的使用又は引用等著作権法上認められた行為として、適宜の方法により出所を明示することにより、引用・転載複製を行うことができます。内容の全部又は一部について、ロボット革命・産業 IoT イニシアティブ協議会に無断で改変を行うことはできません。

ロボット革命・産業 IoT イニシアティブ協議会はいかなる目的においても使用可能性を保証するものではなく、本文書の内容を使用したいかなる場合においても責任を負いません。本文書の使用者は、本文書に記載された内容の使用に関連して発生したすべての要求、請求、訴訟、損失、損害（人身事故による損害を含む）、費用、経費（弁護士費用を含む）について、ロボット革命・産業 IoT イニシアティブ協議会に何らの損害も与えないことに同意するものとします。

目次

1. はじめに.....	4
1.1. ライセンス.....	6
1.2. 参考文献.....	6
2. インストール方法.....	7
2.1. Enterprise Architect へのインストール方法.....	7
2.1.1. SafeML プロファイルのインストール	7
2.1.2. ツールボックス への SafeML 要素の表示	8
3. SafeML の要素.....	10
3.1. 「方策」の関係性.....	10
3.2. 方策に対する責任範囲について	11
4. SafeML 支援ツール	12
4.1. Enterprise Architect による一覧表示.....	12
4.1.1. UserRequirement と Non-Safety Related Function を一覧表示する SQL	15
5. SafeML 使い方事例.....	16
5.1. リスクアセスメントとの関係性：プロセス	16
5.2. ケーススタディ	19
6. 改訂履歴.....	31

1. はじめに

SafeML とは、システムの安全情報を記述するためのモデリング言語で、SysML プロファイルとして定義されています。本書では、モデリング言語 SafeML の使用方法を解説します。

SafeML は、Fault Tree Analysis や HAZOP 等のリスク・安全分析の結果をシステムモデル内に記述するためのモデリング言語です(これらの安全分析を自動で行うためのモデリング言語ではありません)。

SafeML を利用することで、システム情報と安全情報を統合した単一のモデルを作成することができ（図 1）、システムの安全に関する情報の一貫性とトレーサビリティを改良することが可能となります。SafeML は、主に以下の目的で使用する事が可能です。

1. リスク分析などで抽出したハザードと、その安全対策のトレーサビリティの確保
2. 分析を行ったハザードの内容およびその安全対策を認証機関向けに文書化
3. 安全分析で安全エンジニアが抽出したハザードの内容を、要求分析を行うシステムエンジニアに伝達
4. システムエンジニアが検討した安全対策の内容を安全エンジニアに伝達

SafeML は SysML と一緒に利用する事を想定しています。SysML を用いてシステム設計情報をモデル化するとともに、SafeML を用いて安全情報を同じモデル内で表現する事が可能です。SafeML と SysML の関係を図 2 に示します。

SafeML は、モデルベースでシステム設計を行うどのような開発プロセスでも利用することができます。例えば、ウォーターフォール開発でシステム要求の分析後に安全分析を行っているような場合でも、システム要求を SysML で記述し、安全分析の結果を SafeML で追記する事が可能です。更に、システム設計の内容を SysML で作成しながら、安全機能を SafeML で記述することも可能です。開発プロセスについては、章 5.1 で説明しています。

今回の Version 2.0 では、産総研で作成した SafeML の言語仕様、プロフィール、ドキュメント等を元に、RRI-WG3 ソフトウェアアーキテクチャ調査検討委員会内の安全 SA 小 WG にて要素の追加等を実施しました。

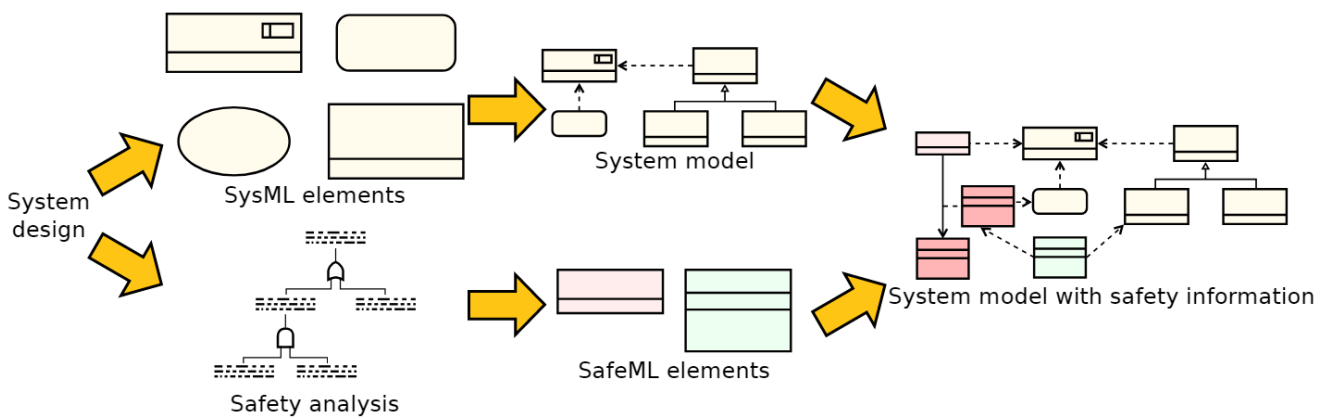


Figure 1 : SafeML のコンセプト

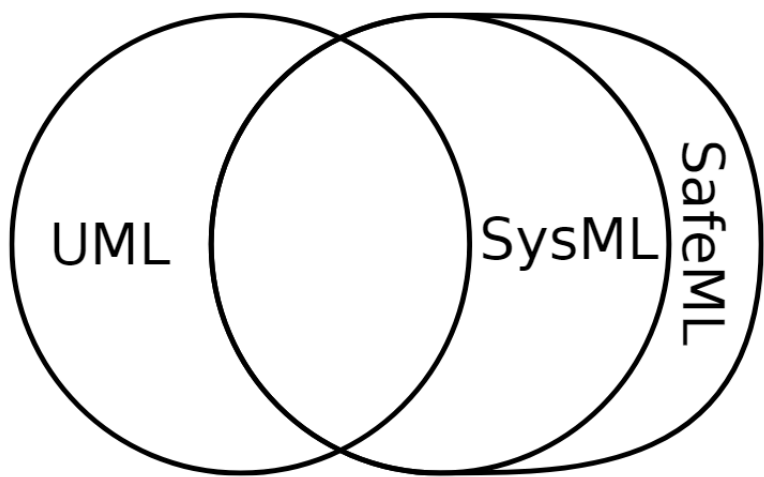


Figure 2: SafeML と SysML の関係

1.1. ライセンス

SafeML のプロファイルは LGPL-3.0 のライセンスで配布します。

1.2. 参考文献

1. “信頼性システム モデリング言語「SafeML」 メタモデル仕様書 Version 2.0,” 2022. <https://www.jmfrri.gr.jp/document/library/2870.html>(2022.06 will be released).
2. “ISO 12100:2010 Safety of machinery — General principles for design — Risk assessment and risk reduction,” <https://www.iso.org/standard/51528.html> (accessed Oct. 03, 2020).
3. 三好崇生, ジェフビグス, and 木村哲也, “人の身近で軽作業に従事するサービスロボット安全の SysML + SafeML 分析,” ロボティクス・メカトロニクス講演会講演概要集, vol. 2018, pp. 2A2–B13, 2018.
4. “JIS Z 8051:2015 安全側面—規格への導入指針 Safety aspects -- Guidelines for their inclusion in standards,” https://webdesk.jsa.or.jp/books/W11M0090/?bunsyo_id=JIS%20Z%208051:2015 (accessed Apr. 20, 2022).

2. インストール方法

2.1. Enterprise Architect へのインストール方法

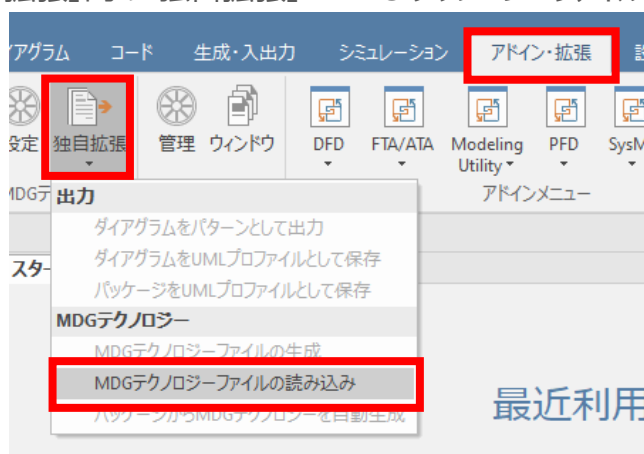
モデリングツール Enterprise Architect を用いて、SafeML モデルを作成する際に必要な事前準備について説明します。

(以下の説明では、執筆時点(2022 年 4 月 19 日)で最新版 Ver.15.2.1560 を使用しています)

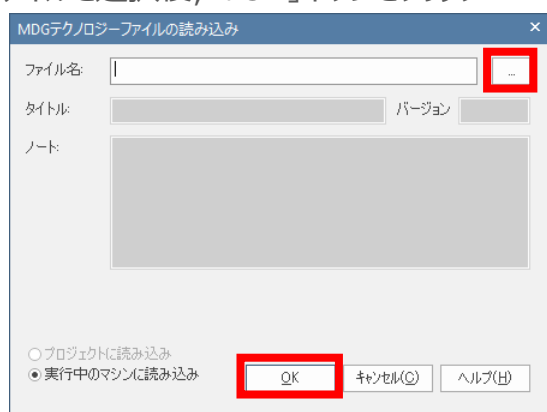
2.1.1. SafeML プロファイルのインストール

SafeML プロファイルとは、SafeML メタモデルで定義された各種要素を Enterprise Architect で使用できるようにするための拡張定義です。以下の手順でインストールを行います。

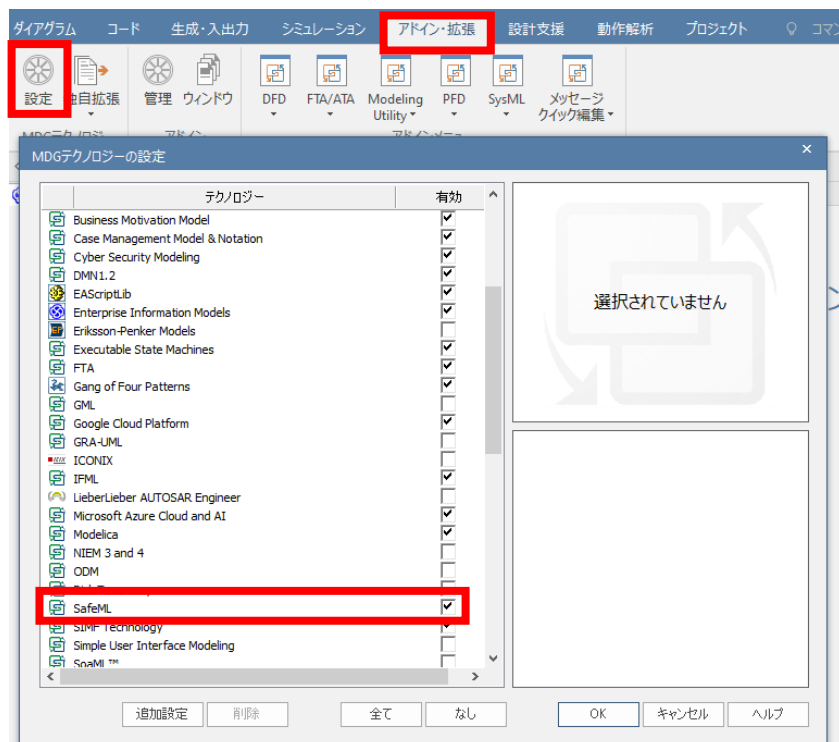
1. 上部メニューの「アドイン・拡張」内の「独自拡張」-「MDG テクノロジーファイルの読み込み」を選択。



2. 表示された【MDG テクノロジーファイルの読み込み】ダイアログ内で、「ファイル名」の左側の「...」ボタンをクリックし、SafeML プロファイルを選択後、「OK」ボタンをクリック



3. 上部メニューの「アドイン・拡張」内の「設定」を選択して表示される【MDG テクノロジーの設定】画面で「SafeML」が追加されていれば、SafeML プロファイルは正常にインストールされています。

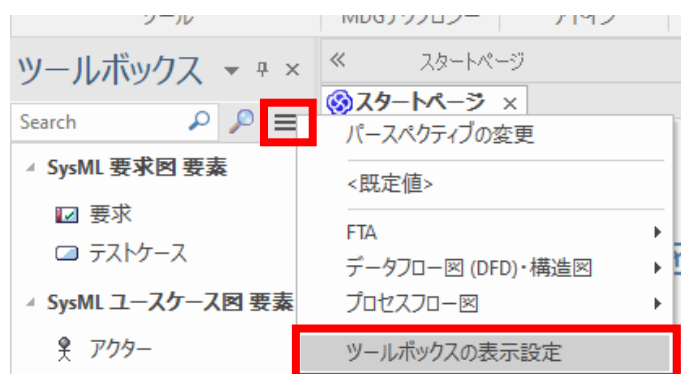


なお、インストールした SafeML プロファイルをアンインストールしたい場合には、【MDG テクノロジーの設定】画面で「SafeML」を選択した状態で、下部の「削除」ボタンを選択してください。

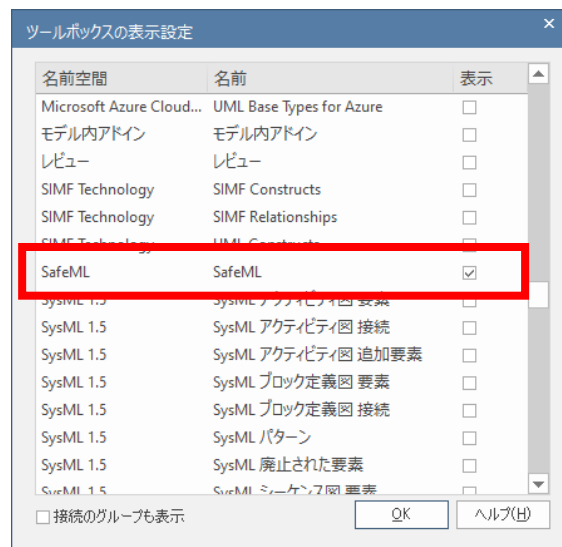
2.1.2. ツールボックス への SafeML 要素の表示

SafeML の要素をツールボックス上に表示するためには、以下の設定を行います。

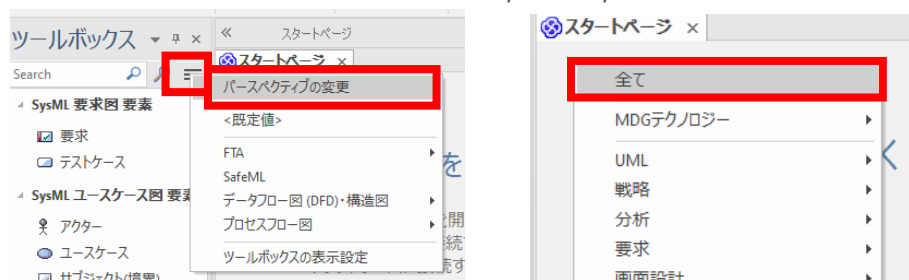
1. ツールボックス内の検索エリア横のボタンを選択し、表示されたメニューの中から、「ツールボックスの表示設定」を選択



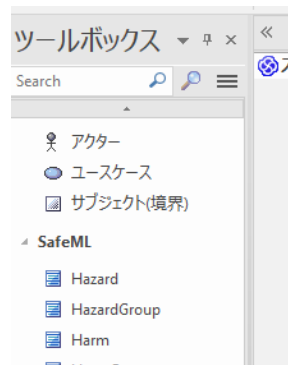
2. 表示された【ツールボックスの表示設定】ダイアログ内で「SafeML」の「表示」列のチェックボックスを ON に設定。



※【ツールボックスの表示設定】ダイアログ内に「SafeML」が表示されない場合には、ツールボックス内の検索エリア横のボタンを選択し、表示されたメニューの中から、「パースペクティブの変更」を選択し、更に表示されたメニュー内から「全て」を選択後、再度、試してみてください。



3. ツールボックス内に SafeML で定義された要素が表示されるようになります。



3. SafeML の要素

要素詳細については、参考文献 1 を参照ください。

3.1. 「方策」の関係性

SafeML 要素はリスクアセスメントをモデル化しているため、リスクアセスメント（参考文献 2 に従う）の方策（対策）に該当する要素がステレオタイプとして定義されています。表 3.1-1 に機械安全で使われるスリーステップメソッド等の安全方策と、SafeML 定義（ステレオタイプ）の比較表を示します。

表 3.1-1 安全方策とステレオタイプの比較表

N o.	SafeML 要素	スリーステップ（設計者の保護方策）			使用者の 保護方策
		本質的	安全防護及び 付加保護	使用上の情報	
1	Passive	○	○ 非機能安全	-	-
2	Active	-	○ 機能安全	-	-
3	Undefence 何もしない	-	-	△	-
4	ProActive 動的安全	-	○ 非機能安全	○	○

SafeML2.0 で「ProActive Defence」が拡張されましたが、同時に Passive や Active の意味を明確にするため、従来の機械安全で示された方策との対応表を作成しました。

注意点として「Active」のみ機能安全による対策を受け持ち、「ProActive」ではそれら以外のソフトウェア的な対応が含まれるようにしています。「Passive」での安全防護については柵等が含まれます。

3.2. 方策に対する責任範囲について

SafeML2.0 で追加された新要素であり、事例は 5 章で紹介しています。

方策に対する責任範囲は従来の安全設計においても関係性を明確にすることが求められていますが、SafeML2.0 ではモデル要素を使って方策と責任者の関係性を示す手法を提案しました。これにより、従来の SafeML よりもより安全設計に適したモデルが作成できるようになりました。

4. SafeML 支援ツール

5 章で、安全設計プロセスにおける SafeML の位置づけを示しますが、作業としては SafeML のみで完結せず他の作業から情報を引き継ぎ、成果を渡す、という Input-Logic-Output の関係が成り立ちます。ここでは、SafeML から Output する作業方法について示します。

4.1. Enterprise Architect による一覧表示

Enterprise Architect は強力な検索機能を持っており、2 次元の一覧表を作成することが可能となっています。検索機能の立ち上げは、「Ctrl+F」もしくは、メニューから選択することで立ち上げが可能です。

検索機能はいくつかのモードを持っていますが、SQL を直接編集して情報を取得する機能も有しています。Enterprise Architect はモデルを RDB 形式で保管しており、モデリング作業とは RDB に追加、修正、削除を行っていることになります。

検索においても内部データの構造（SafeML がどのようなリレーションで RDB に保存されているか）が分かれば、SQL 文を作成し出力することが可能となります。ただ、内部構造はツールベンダでは、サポート範疇外とされているようです。ここでは、わかる範囲の情報をまとめ、検索の実際の SQL 事例を示したいと思います。（そのため、間違いがあるかもしれませんが、適宜読み替えをお願いします）

図 4.1-1 に新規の検索ルールの作成（赤枠内）を示します。



図 4.1-1 新規の検索ルールの作成（赤枠内）

クリックすると、図 4.1-2 検索ルールの新規作成ダイアログが表示されます。

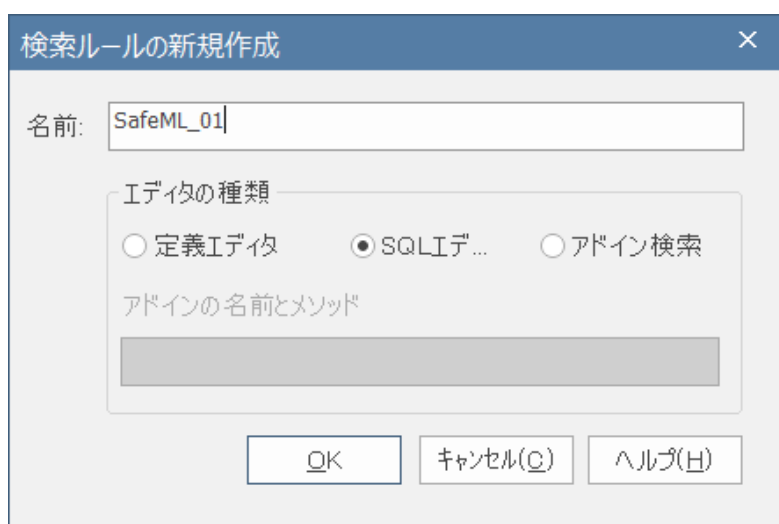


図 4.1-2 検索ルールの新規作成ダイアログ

SQL エディタにラジオボタンを切り替え、名前は区別のつくものとして、OK をクリックしてください。すると、図 4.1-3 SQL エディタ画面が表示されます。（ここでは、SQL 入力欄に aaaa と適当な文字列を記入しています）

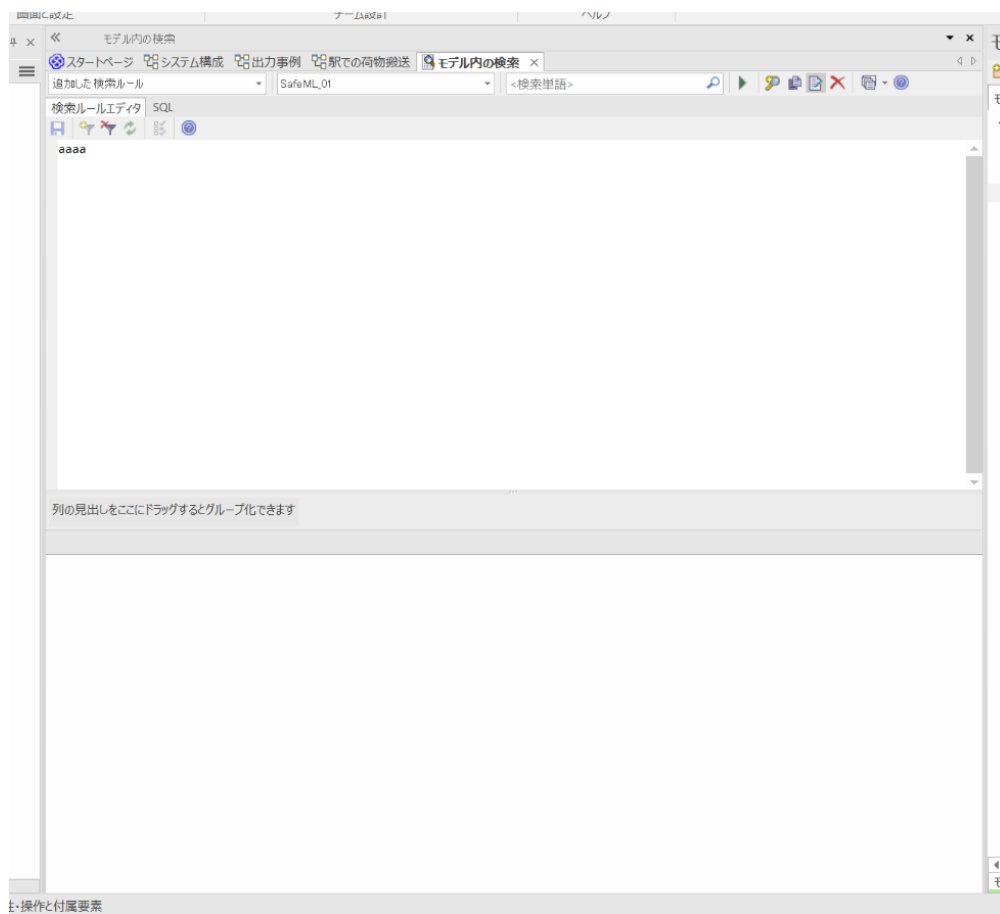


図 4.1-3 SQL エディタ画面

4.1.1. UserRequirementと Non-Safety Related Functionを一覧表示する SQL

図 4.1.1-1 に SQL 構文例を示します。

```
SELECT
    t_object.ea_guid AS CLASSGUID,
    t_object.Object_Type AS CLASSTYPE,
    t_object.Name,
    t_object.Stereotype
FROM t_object
WHERE (
    ((t_object.Stereotype)="Non-Safety Related Function"))
OR
    (((t_object.Stereotype)="UserRequirement"));
```

図 4.1.1-1 SQL 構文例

Enterprise Architect 独自のルールとして、SELECT 内の 2 行 ea_guid/object_type があります。これらを付加することで、検索結果をダブルクリックすることでプロパティの編集画面に移動することができます。

5. SafeML 使い方事例

ここでは、SafeMLと既存の機械安全（参考文献 2）での考え方を比較しながら使い方を説明します。そのため読者として安全設計の基礎知識がある方を対象としました。また、Enterprise Architect 15以降での SysML での作業についてもある程度の経験がある方を対象としています。（例えば、ツールベンダが提供している「SysML 操作セミナー」を受講したことがあり、基礎操作ができる方）

5.1. リスクアセスメントとの関係性：プロセス

ガイド 51（参考文献 4）や ISO12100（参考文献 2）等で見られる通り、安全設計にはプロセスが定められています。これら安全設計プロセスと機能設計プロセスが並行に処理され、一つのシステム（サービスロボット）が創られるものと考えます。

プロセスの上位に存在する考え方として、スリーフェーズ開発も一般的に行われています。これは、試作、実証実験、製品化といったフェーズ分けされた考え方となり、設計プロセスはフェーズ内で繰り返し実施されます。

これら考え方を模式図で表したものが、図 5.1-1 となります。

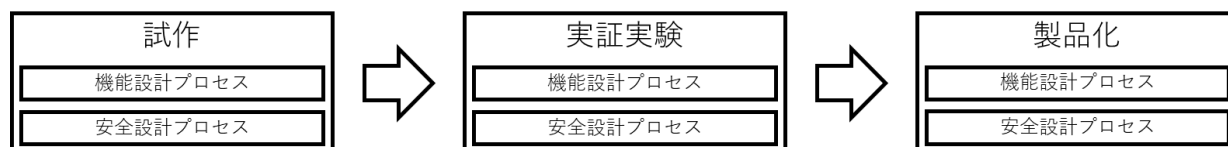


図 5.1-1 スリーフェーズと安全／機能設計プロセスの関係性

SafeML をスリーフェーズ等の複数フェーズが存在する開発プロセスで使う事例を考えます。例えば試作の安全設計プロセスの中で使われますが、次フェーズ以降（実証実験や製品化）においても有効に設計が活用できるよう再利用性があることが求められます。ここで一般的に言われることですが、SafeML を含む MBSE と呼ばれる手法は、再利用性が高いと言われています。これら特徴は、様々な研究がなされていますが、ヒューリスティックな評価とならざるを得ないところでもあり、事例ごとに合理的な判断も必要になると思われませんが、ここでは試作フェーズも含む全ての安全設計プロセスで利用可能であろうと考えます。

では、具体的に安全設計プロセス内を見ていきましょう。ここでは、ガイド 51（参考文献 4）から、プロセスを参照しました。図 5.1-1 にガイド 51 の安全設計プロセスを示します。

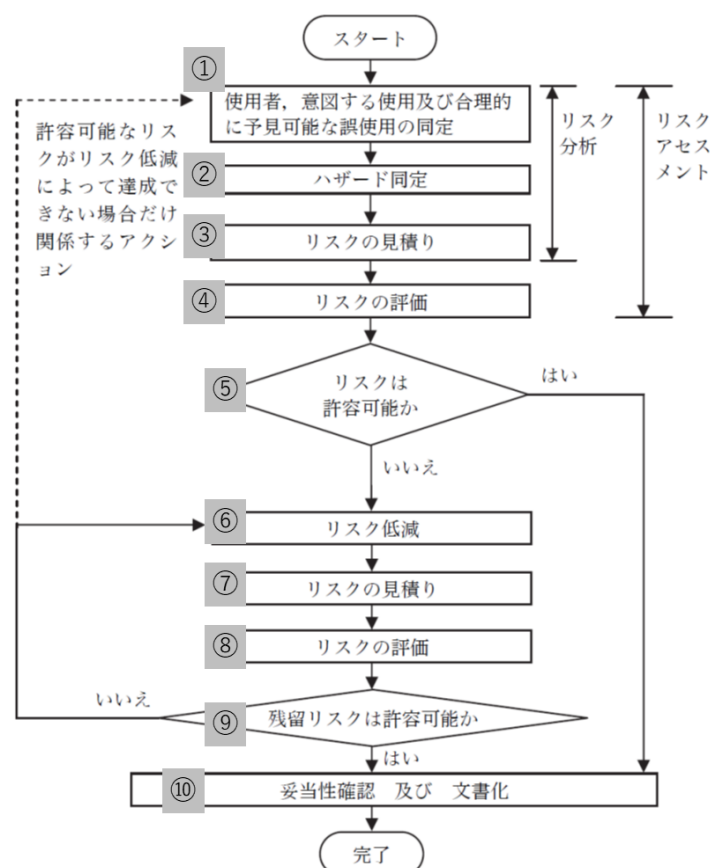


図 5.1-1 ガイド 51 安全設計プロセス

一般的なリスクアセスメント表における作業プロセスは、表作成の前段階として誤使用等を検討（図中①）し、次に各個別の行単位でのリスク抽出に移る形になります。SafeML でもこれらの流れは大きくは変わりません。ただ、①の作業や⑩文書化等では SysML により、設計上のメリットが出てきます。SafeML は SysML の拡張のため、ベースの SysML が持つ特性が安全設計時にもメリットに繋がります。

一つの例として、情報の定義と利用が分けられているという特徴から説明します。従来の表形式のリスクアセスメントにおいて①を定義する場合、自然言語を利用し表内の定められた場所に記述（定義及び利用）を行っていました。これは、再利用性を下げるものであり、一般的に再利用を行う上では構文解析（人や AI による）を行い、意味を持った情報に分解し、必要な形に再構築する、というプロセスが必要となります。実際に、安全設計時に①を抽出した設計者は、意味を理解した上で抽出しているため、この設計手法では構文解析を異なる設計者が 2 度行う、という無駄を発生させています。この課題に対し、SysML や SafeML では、ステレオタイプにより意味を持った要素として設計情報を管理しています。（ただし、設計情報の細分化する粒度は SafeML の言語仕様にも依存します）これにより、定義と利用が情報として管理されるようになり、個別の抽出が可能となります。このような特徴はトレーサビリティが要求されるような開発においても有効と言えます。

SafeML では、①の条件等は機能設計プロセスで設計されたモデル要素を参照することが可能です。これは従来のリスクアセスメント表よりも自由度が高く、ある一行のリスクに対しても細かく設定することが可能となっています。

これらのことから、SafeML 活用が従来の安全設計プロセス内にメリットをもって組み込めることが確認できました。ただ、実際に使う上では、作成にかかる手間がどの程度か、という課題もあります。表形式は、一定の情報を順次入力する場合にはとても有効な作業インターフェースです。一方の SafeML モデルの作成は、モデルの見た目を整える（ブロック要素の並びを揃える、サイズを変更する）作業が、設計作業中にも

発生します。これらについては、5.2 章のケーススタディを確認いただき、各設計者の設計物において見せたい点を明確にした上でのご活用をお勧めします。なお RRI-WG3 ソフトウェアアーキテクチャ調査検討委員会 安全 SA 小 WG においては、人との関係性があるもの、ステークホルダが多い事例等で、SafeML の利便性が示しやすいと考えています。

5.2. ケーススタディ

表 5.2-1 のリスク事例をモデル化する流れを順次確認していきます。

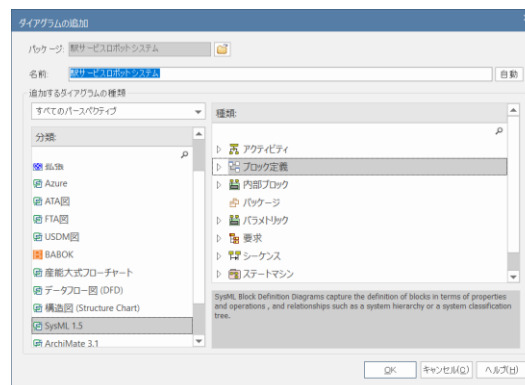
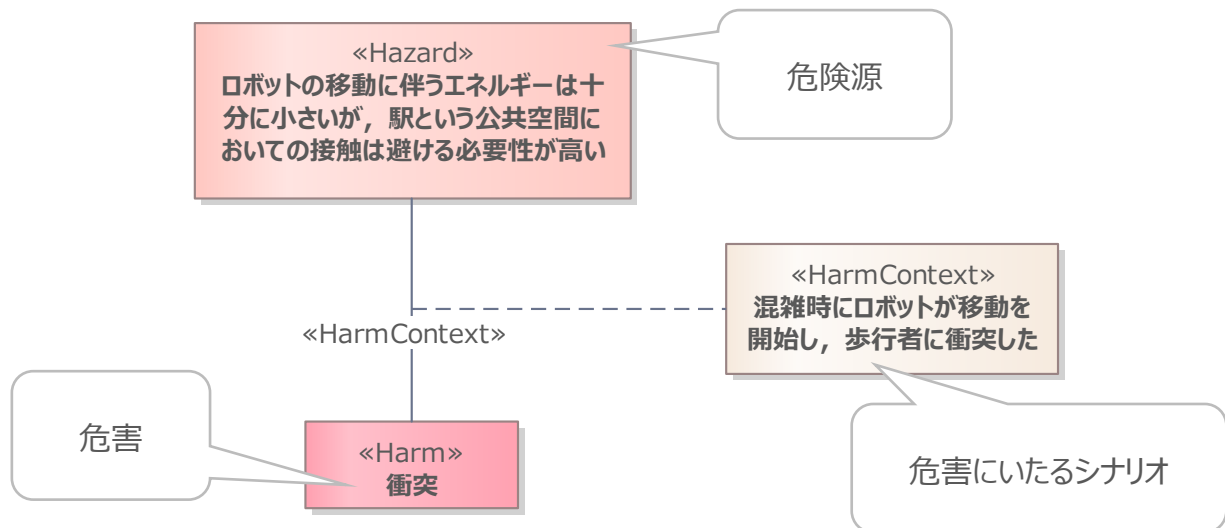
事例では、人流に動的な変化が起こる環境で、ロボットがどのように動くことが安全か、との検討を行います。人が多い場合、ロボットが退避する等の対策で回避可能性が上昇しますが、複数のシステムが関係する事例のため、ロボット内部の安全関連部で実装するものだけではなく、複数のシステムが協力し実現する方策も取り込みモデル化を行っていきます。このような複雑さを俯瞰的に見せ、SafeML の利点も確認できる事例となっています。

表 5.2-1 リスク事例

前提条件	人の量が増える駅改札前で、荷物搬送を行うロボットの周囲の人の変化に伴う RA である。 小型軽量、低速なロボットであり、衝突そのものは低いリスクではある。		
Hazard	Harm	Harm Context	Defence
駅という公共空間において、ロボットと人の接触は避ける必要性が高い	人とロボットの衝突に伴う、二次災害	混雑時にロボットが移動を開始し、人に衝突した	人が多いときには、ロボットは退避エリアにて待機状態となる

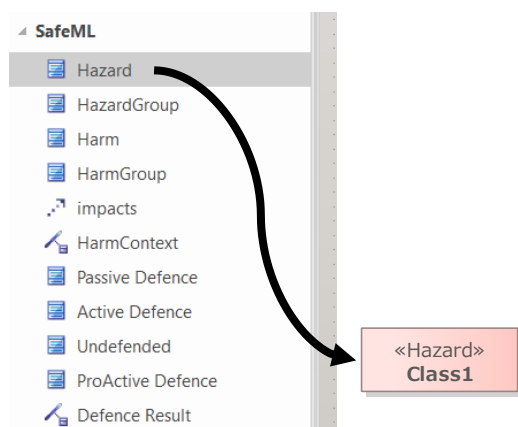
図 5.1-1 の②～⑤の作業（リスクアセスメント）

ここで作成する図は以下のリスクアセスメント基本モデルとなります。



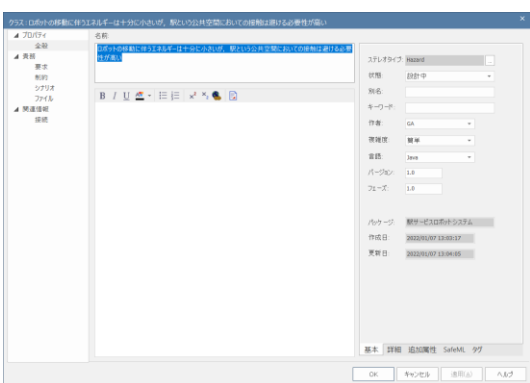
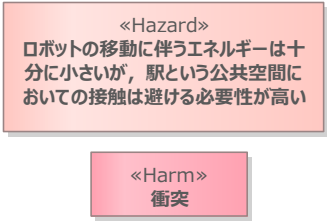
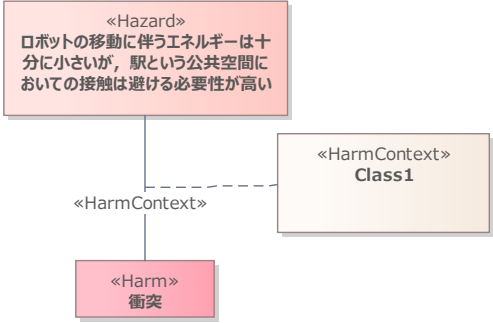
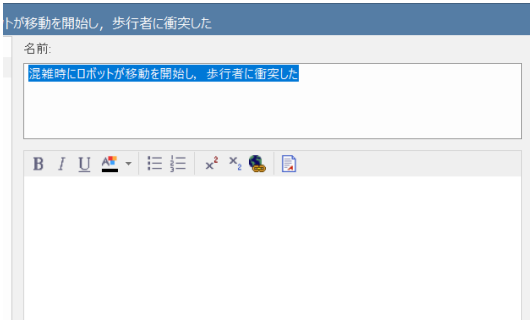
「SysML1.5」内の「ブロック定義」を作成します。

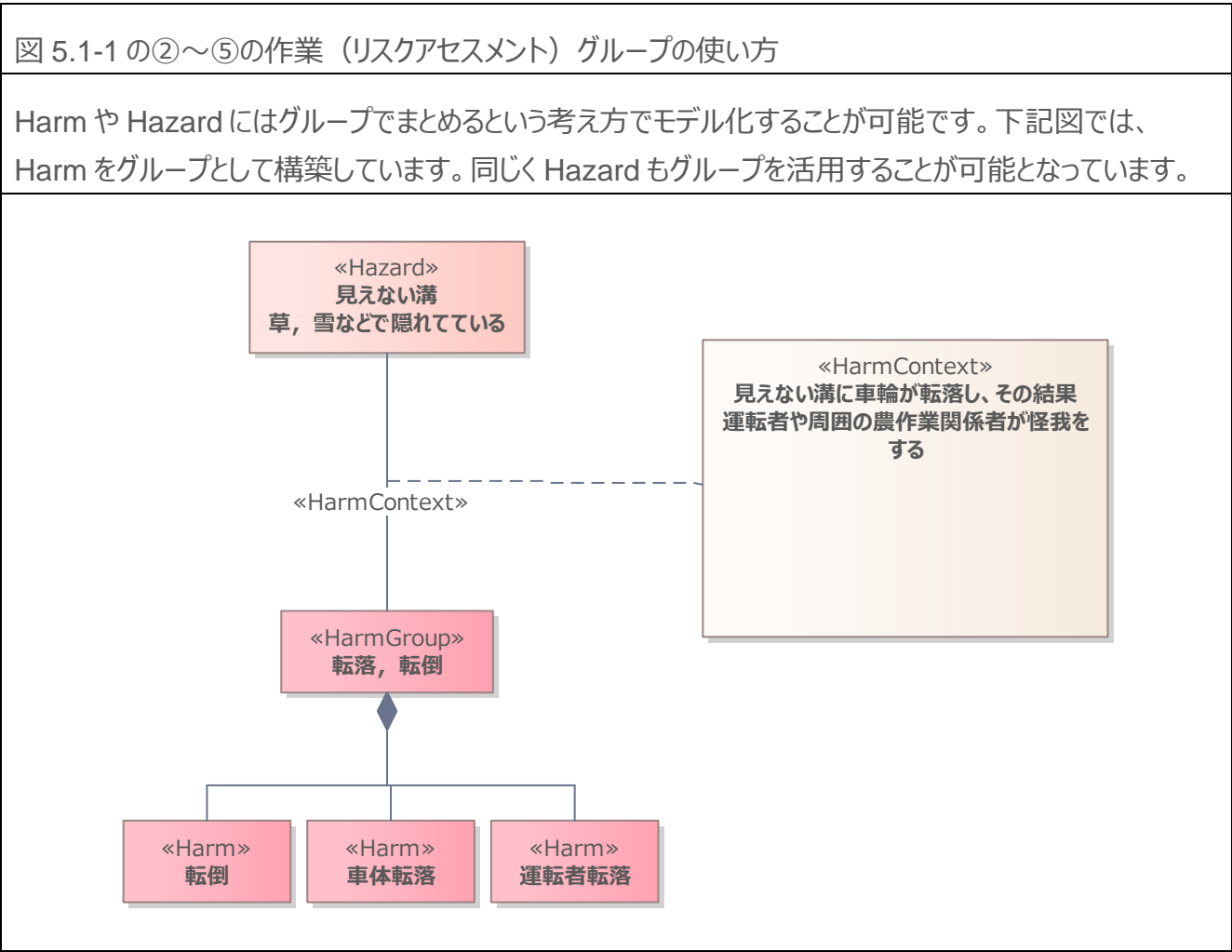
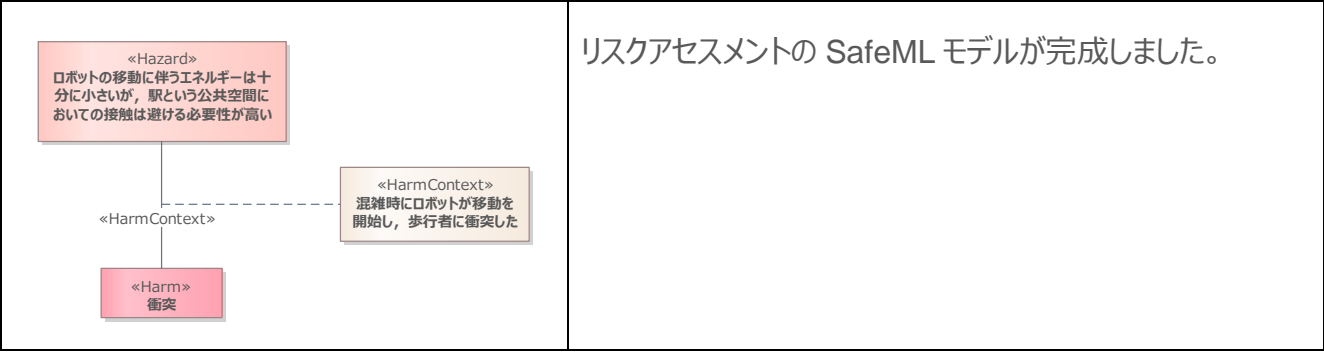
ダイアグラム名は、リスクアセスメントのユースケースとなる「駅での荷物搬送」としています。


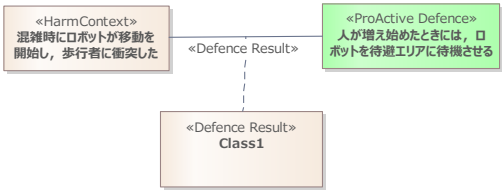
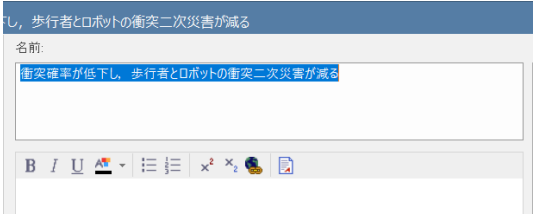
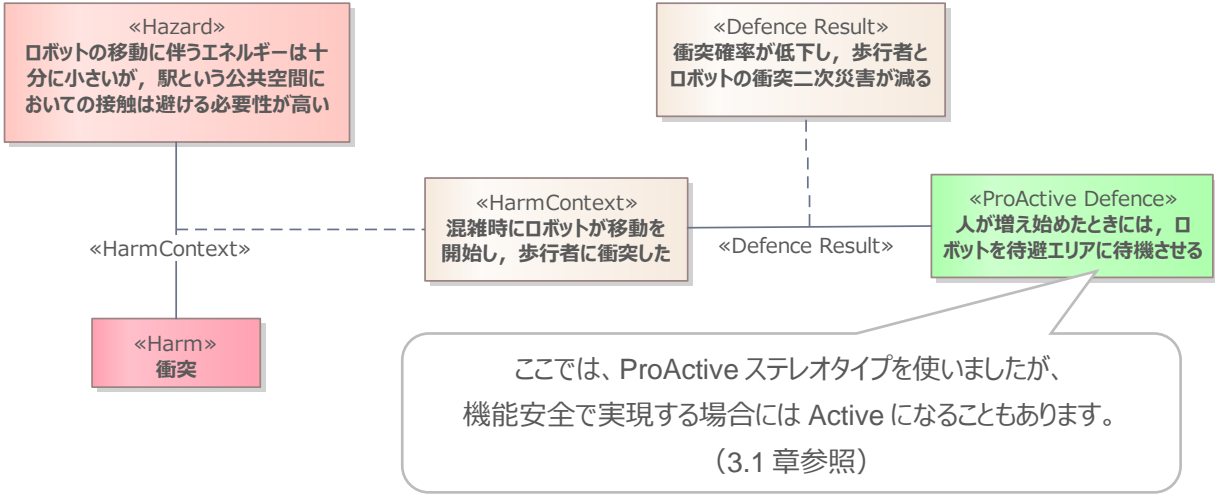


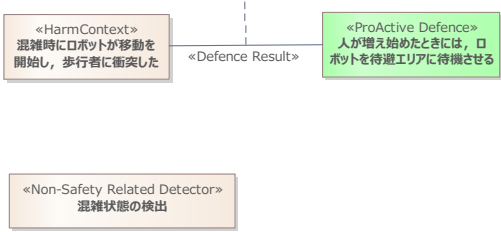
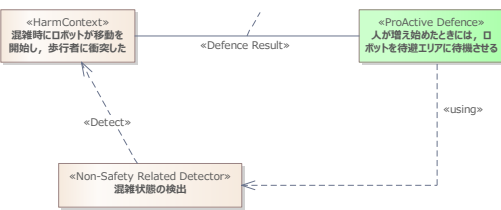
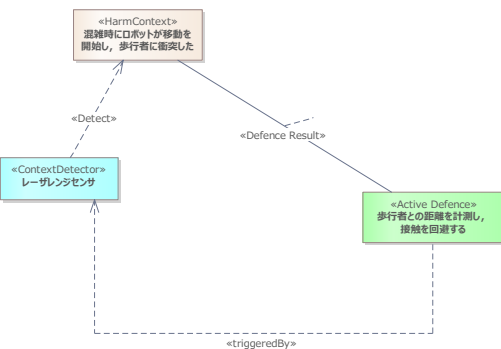
次に要素を貼り付けます。

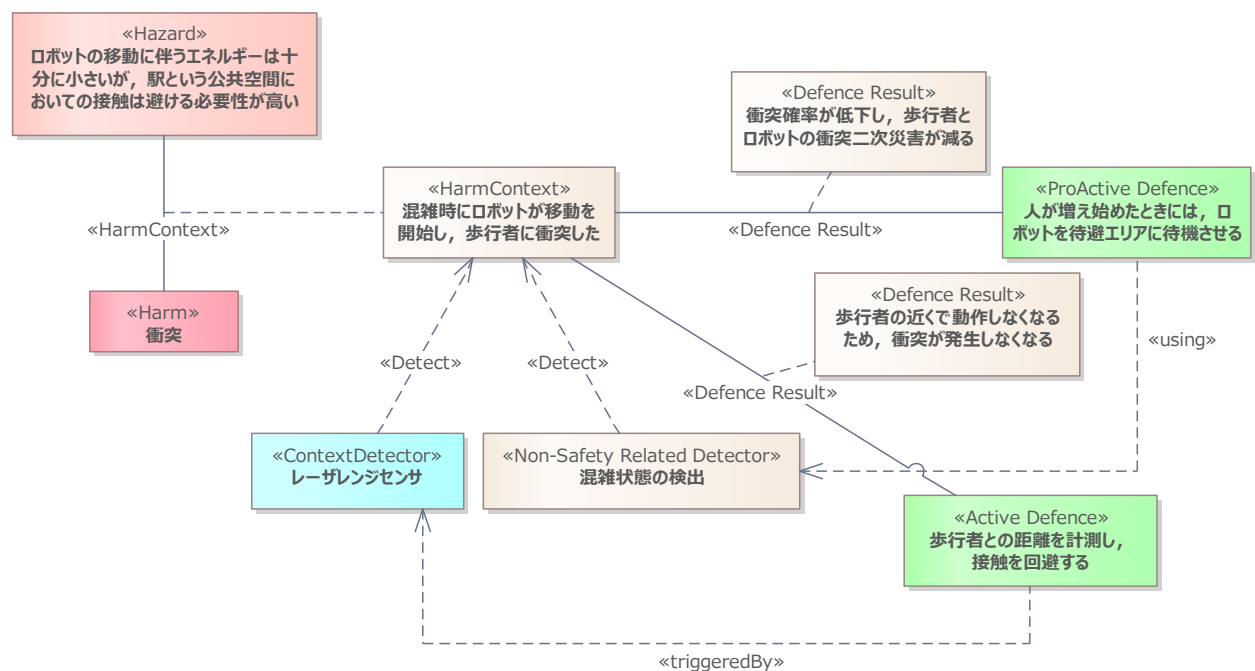
SafeML のツールボックスから、「Hazard」を選択し、ダイアグラムの上でクリックすると、何も記入されていない«Hazard»要素が配置されます。

	<p>«Hazard»要素をダブルクリックし、名前欄に Hazard 内容を記入します。</p>
<p>-</p>	<p>«Harm»も同様の作業を行います。</p>
	<p>«Hazard»と«Harm»が準備できました。</p>
	<p>次にツールボックスの「HarmContext」を選択します。「HarmContext」は2つの要素間にまたがって関連ブロックが生成される要素となっています。</p> <p>«Hazard»から«Harm»に線を引くと、«HarmContext»が作成されます。（Hazard→Harm でも Harm→Hazard でも問題なく作成できます）</p>
	<p>«HarmContext»要素をダブルクリックし、名前欄に記入します。</p>

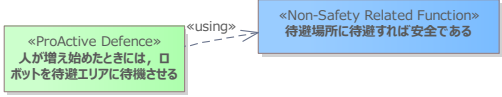
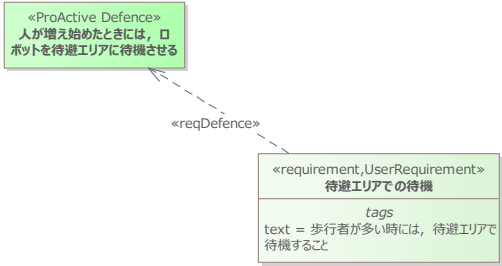
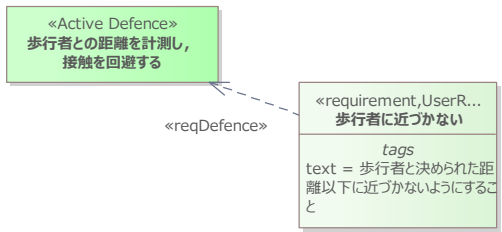
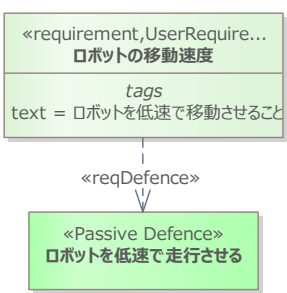
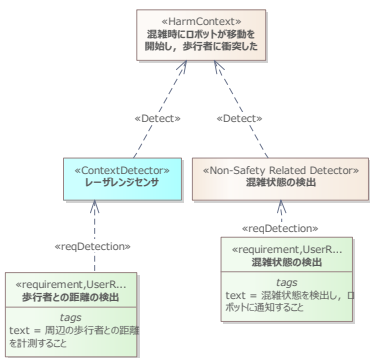


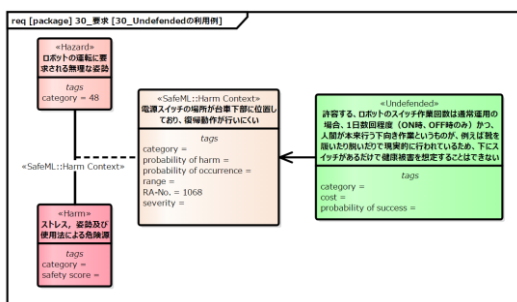
<p>図 5.1-1 の⑥～⑨の作業（方策のモデル化）</p>	<p>Defence の使い方区分については、3.1 章を参考にしてください。ここでは、ProActive 要素の作成方法を紹介します。</p> <p>ツールボックスの「ProActive」を選択し、ダイアグラム上に配置します。</p>
	<p>名前欄に Defence 内容を記入します。</p>
	<p>ツールボックスの「Defence Result」を選択します。</p> <p>次に、「«ProActive Defence»」から「«HarmContext»」に線を引くと、「«Defence Result»」が作成されます。</p>
	<p>名前欄に Defence Result 内容を記入します。</p> <p>ここでは、方策の結果、よりリスクが小さくなる事を示しています。</p> <p>作成できたモデルを以下に示します。</p>
	

 <pre> classDiagram class HarmContext["«HarmContext» 混雑時にロボットが移動を開始し、歩行者に衝突した"] class ProActiveDefence["«ProActive Defence» 人が増え始めたときには、ロボットを待避エリアに待機させる"] class NonSafetyRelatedDetector["«Non-Safety Related Detector» 混雑状態の検出"] HarmContext --> ProActiveDefence : «Defence Result» </pre>	<p>次に、「HarmContext」を検出する機能を追加します。</p> <p>Defence の違いによって、使える検出は異なります。</p> <p>「ProActive Defence」の場合には、「Non-Safety Related Detector」を利用します。</p> <p>「Active Defence」の場合には、「Context Detector」を利用します。</p>
 <pre> classDiagram class HarmContext["«HarmContext» 混雑時にロボットが移動を開始し、歩行者に衝突した"] class ProActiveDefence["«ProActive Defence» 人が増え始めたときには、ロボットを待避エリアに待機させる"] class NonSafetyRelatedDetector["«Non-Safety Related Detector» 混雑状態の検出"] HarmContext --> ProActiveDefence : «Defence Result» NonSafetyRelatedDetector -.-> HarmContext : «Detect» NonSafetyRelatedDetector -.-> ProActiveDefence : «using» </pre>	<p>以下、矢印の向きに注意が必要です。（アローがつくのは、後にドラッグした側となります）</p> <p>「ProActive Defence」と「Non-Safety Related Detector」の間を「using」でつなぎます。</p> <p>「Non-Safety Related Detector」と「HarmContext」の間を「Detect」でつなぎます。</p>
 <pre> classDiagram class HarmContext["«HarmContext» 混雑時にロボットが移動を開始し、歩行者に衝突した"] class ContextDetector["«ContextDetector» レーザレンジセンサ"] class ActiveDefence["«Active Defence» 歩行者との距離を計測し、接触を回避する"] ContextDetector -.-> HarmContext : «Detect» ContextDetector -.-> ActiveDefence : «triggeredBy» HarmContext --> ActiveDefence : «Defence Result» </pre>	<p>注記：「Active Defence」の場合</p> <p>「Active Defence」と「Context Detector」の間を「triggeredBy」でつなぎます。</p> <p>「Context Detector」と「HarmContext」の間を「Detect」でつなぎます。</p> <p>作成できたモデルを以下に示します。</p>



一般的なリスクアセスメント表でいうと、リスクアセスメント x 1 + 方策 x 2（ProActive と Active）が抽出できたモデルとなります。ここからさらに、抽出すべき情報として、リスクアセスメント方策をまとめた要求として管理していく必要があります。（以降にて解説します）

図 5.1-1 の⑩の作業（文書化：要求の導出）	
⑩は妥当性確認／文書化を行うプロセスとなりますが、ここでは方策からの要求の導出例を示します。	
 <pre> graph LR A["«ProActive Defence» 人が増え始めたときには、ロボットを待避エリアに待機させる"] B["«Non-Safety Related Function» 待避場所に待避すれば安全である"] A -.-> «using» B </pre>	<p>ProActive では二種類の方法が規定されています。</p> <p>«using»で接続し«Non-Safety Related Function»を設定する場合と«reqDefence»で接続し«UserRequirement»を呼び出す場合の2パターンとなります。</p> <p>前者は「機能」を説明する場合に利用し、後者は「要求」を示すために利用します。</p>
 <pre> graph LR A["«ProActive Defence» 人が増え始めたときには、ロボットを待避エリアに待機させる"] B["«requirement, UserRequirement» 待避エリアでの待機 tags text = 歩行者が多い時には、待避エリアで待機すること"] A -.-> «reqDefence» B </pre>	
 <pre> graph LR A["«Active Defence» 歩行者との距離を計測し、接触を回避する"] B["«requirement, UserR...» 歩行者に近づかない tags text = 歩行者と決められた距離以下に近づかないようにすること"] A -.-> «reqDefence» B </pre>	<p>Active では、«reqDefence»で接続し«UserRequirement»を導出します</p> <p>«using»は利用できません。</p>
 <pre> graph TD A["«requirement, UserRequire...» ロボットの移動速度 tags text = ロボットを低速で移動させること"] B["«Passive Defence» ロボットを低速で走行させる"] A -- «reqDefence» --> B </pre>	<p>Passive では、«reqDefence»で接続し«UserRequirement»を導出します。</p> <p>«using»は利用できません。</p>
 <pre> graph TD A["«HarmContext» 避難時にロボットが移動を開始し、歩行者に衝突した"] B["«ContextDetector» レーザレンジセンサ"] C["«Non-Safety Related Detector» 避難状態の検出"] D["«requirement, UserR...» 歩行者との距離の検出 tags text = 周辺の歩行者との距離を計測すること"] E["«requirement, UserR...» 避難状態の検出 tags text = 避難状態を検出し、ロボットに通知すること"] D -.-> «reqDetection» B E -.-> «reqDetection» C B -.-> «Detect» A C -.-> «Detect» A </pre>	<p>«HarmContext»を検出する«Context Detector»や«Non-Safety Related Detector»についても、«UserRequirement»を導出することになります。</p> <p>ただしこの場合の接続は、«reqDetection」となります。</p>

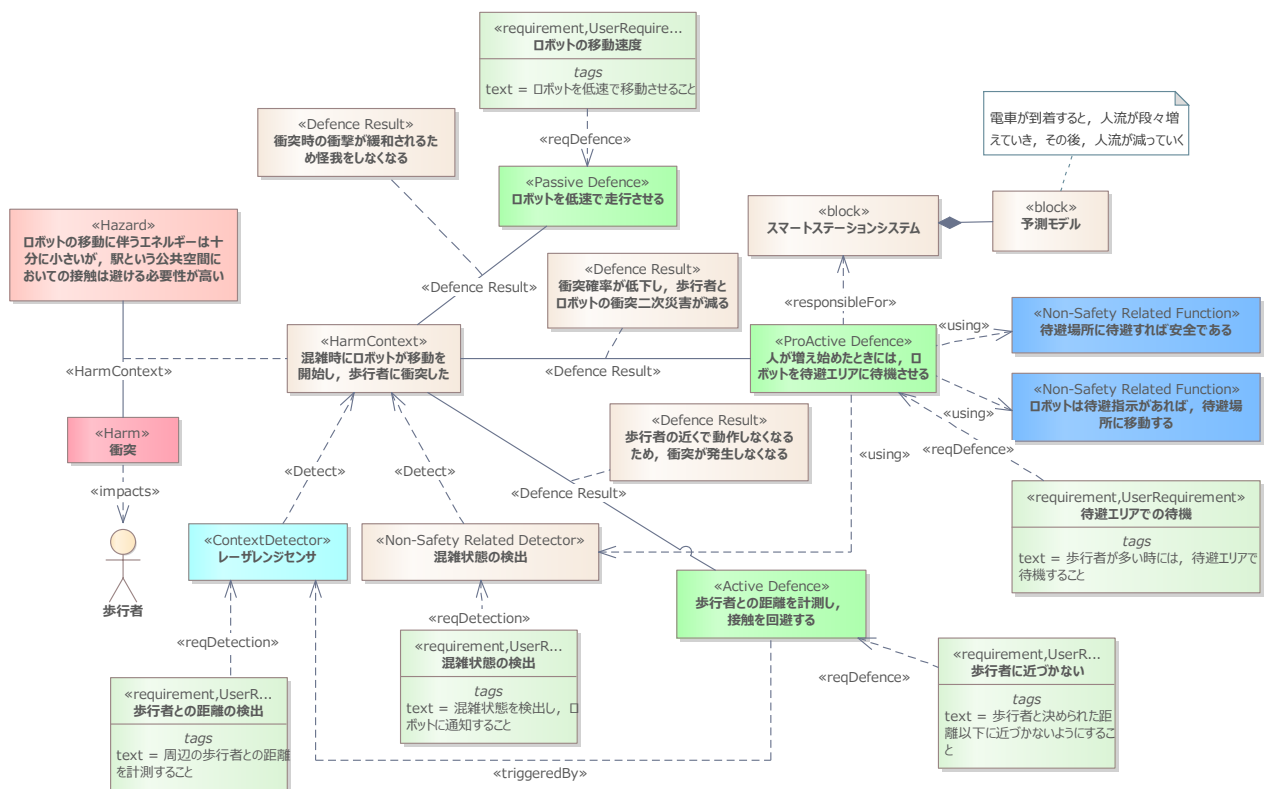


Undefended の場合には、方策の接続は基本的にはありません。

ただ言語仕様としては、「reqDefence」で接続し「UserRequirement」を呼び出すことは可能となっています。

図引用元：参考文献 3

最終的に抽出されたモデル事例と 4.1 章の方法にて抽出した結果を以下に示します。

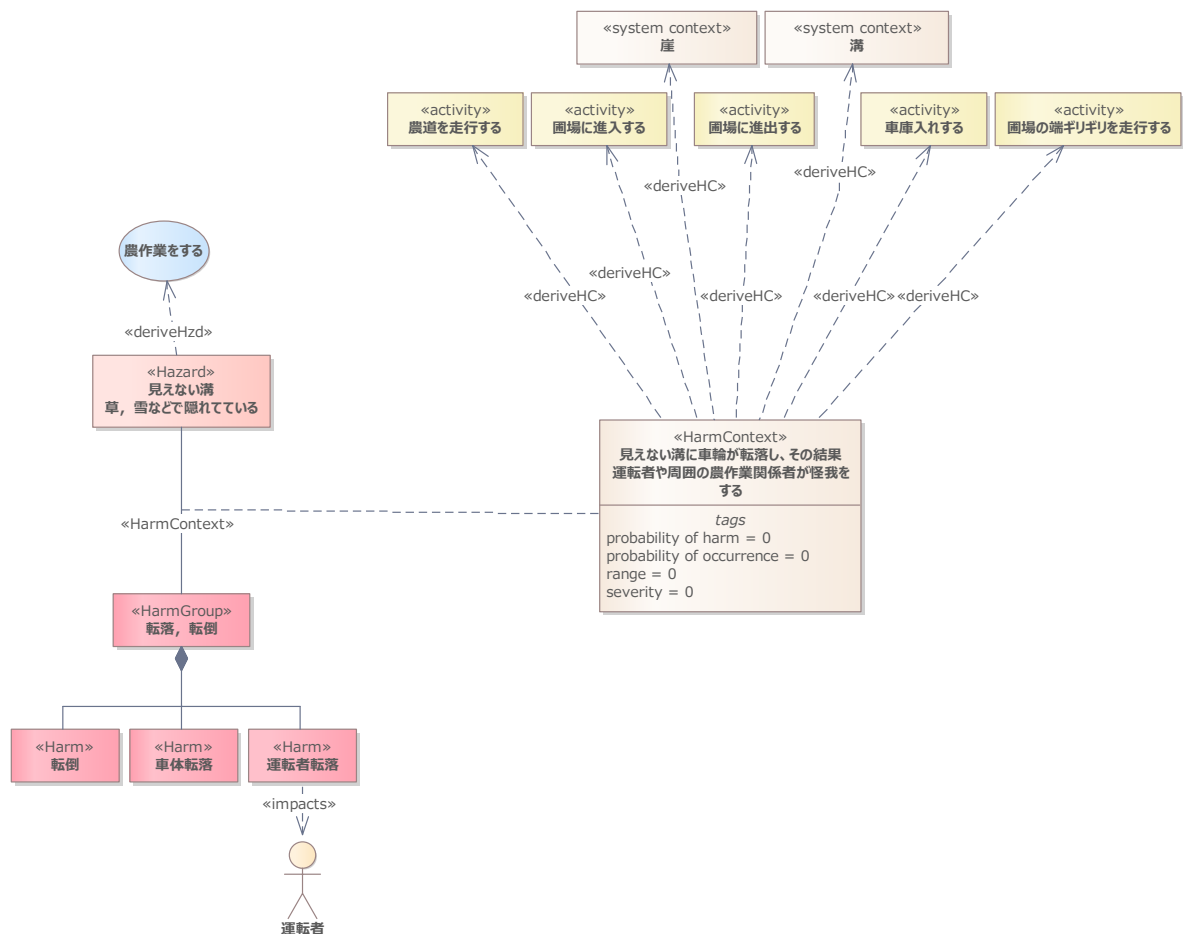


モデル内の検索		
駅での荷物搬送 モデル内の検索 x		
追加した検索ルール	SafeML_01	<検索単語>
列の見出しをここにドラッグするとグループ化できます		
Name	Stereotype	
混雑状態の検出	UserRequirement	
待避エリアでの待機	UserRequirement	
待避場所に待避すれば安全である	Non-Safety Related Function	
ロボットは待避指示があれば、待避場所に移動する	Non-Safety Related Function	
ロボットの移動速度	UserRequirement	
歩行者との距離の検出	UserRequirement	
歩行者に近づかない	UserRequirement	

図 5.1-1 の①の作業（周辺情報のモデル化）

リスクアセスメントの周辺情報をモデル内で呼び出すことが可能です。

ここでは、これまでに参考にしてきた事例と異なり、農機ロボットの事例を示します。（必要か所のみ抜粋したもの）



HarmContext に関係する«activity»を«deriveHC»で関連付けしています。

別のアクティビティ図（振る舞いを示すフロー図）が存在しており、その中の Activity 要素が呼び出されている形式となっています。

また、周辺環境モデルとして«system context»要素も同じく«deriveHC»で関連付けしています。

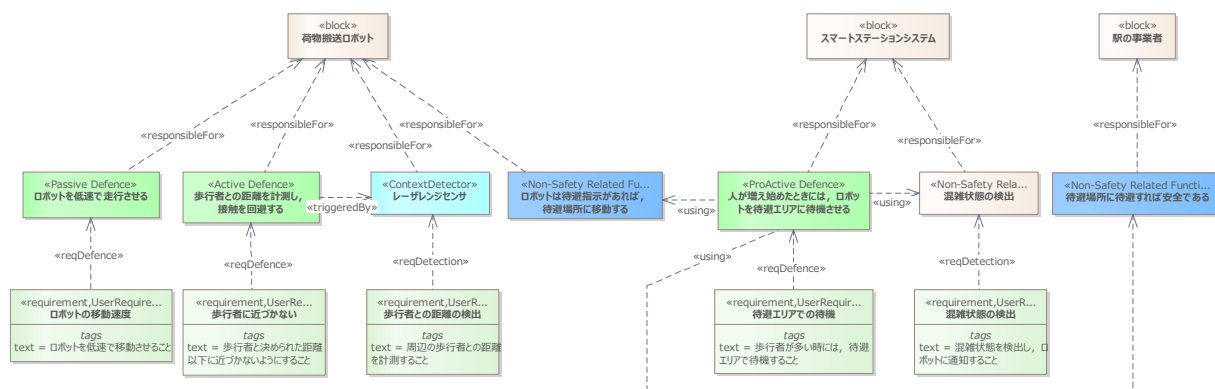
HarmContext から導出される要素を追加していく事で、リスクアセスメントにおける①の作業が意味を持ったモデルとして作成可能となっています。

これら以外に、「Hazard」から導出可能な要素に対し「deriveHzd」にて要素を追加することも可能となっています。（ここでは「農作業をする」というユースケース）

また、Harm から実際に障害を与えるアクターを「impact」にて関連付けすることも可能となっています。

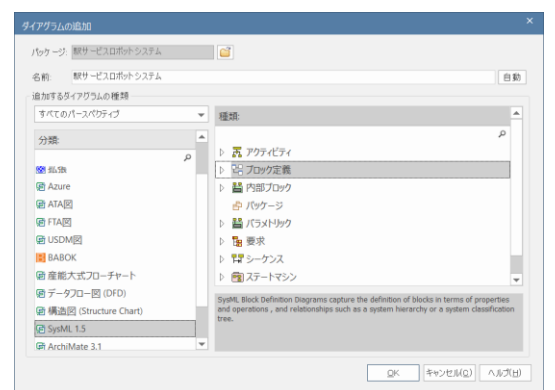
（その他：責任範囲のモデル化）： SafeML2.0 で追加された要素

以下は、「responsibleFor」で関連付けた事例におけるリスクに対しての責任分担を示したモデルとなっています。（ただし、例えばステークホルダとして登場している駅の事業者等と協議して決定したモデルではなく、あくまで仮定のものとなります）


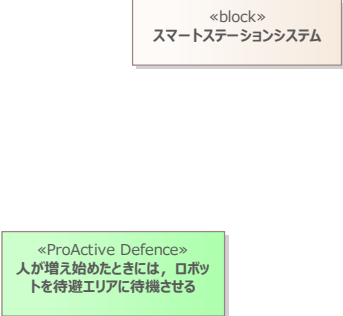
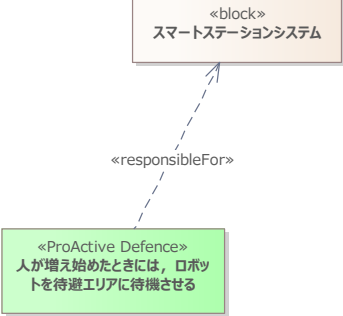


駅の事業者が「responsibleFor」で持つ「Non-Safety Related Function」 「退避場所に～」については、例えば作業当日の退避場所が変更になるようなイメージを持っており、その場所指定を行うことに対しては責任を持つべきではないか、との仮定を行いました。

このような複雑なシステムにおける責任分界点を明確にすることも SafeML の目的の一つとなっています。以下、このモデル作成の手順を示します。



新規のブロック定義図を作成します。

	<p>ブロック要素として、責任を負うべき対象を設計します。</p>
	<p>先に作成した SafeML モデルの要素をコピーし、現在のモデルにペーストします。</p> <p>ここでは、ProActive 要素「人が増え始めたときには～」をペーストしました。</p>
	<p>ツールボックスから responsibleFor を選択し、「ProActive」から責任の対象に線を引きます。（ドラッグによる）</p> <p>矢印の方向がありますので、ドラッグする順番に注意してください。</p>
<p>以下、この作業を漏れなく行い、表示すべき情報等の整理を行うことで、モデルが作成できます。</p> <p>ここでは、元の SafeML 図面からコピー＆ペーストを行いながら作業を行いましたが、実は EnterpriseArchitect 内では、同一のモデルとして管理されています。（モデルブラウザ等で確認を取ってください）</p> <p>同一のモデル要素が異なる設計 View に対応できるという SysML が本来持っている機能を SafeML でも活用し、モデル作成が行えるようになっています。</p>	

6. 改訂履歴

版番号	公開日	備考
2.0	2022/06/01	第 2 版（V1.0 は産総研で公開された）



ロボット革命・産業IoTイニシアティブ協議会
Robot Revolution & Industrial IoT Initiative