Traceability of Model : Example

by SparxSystems Japan

トレーサビリティを活用したモデル構築の例

(2023/06/01 最終更新)

目次

はじめに

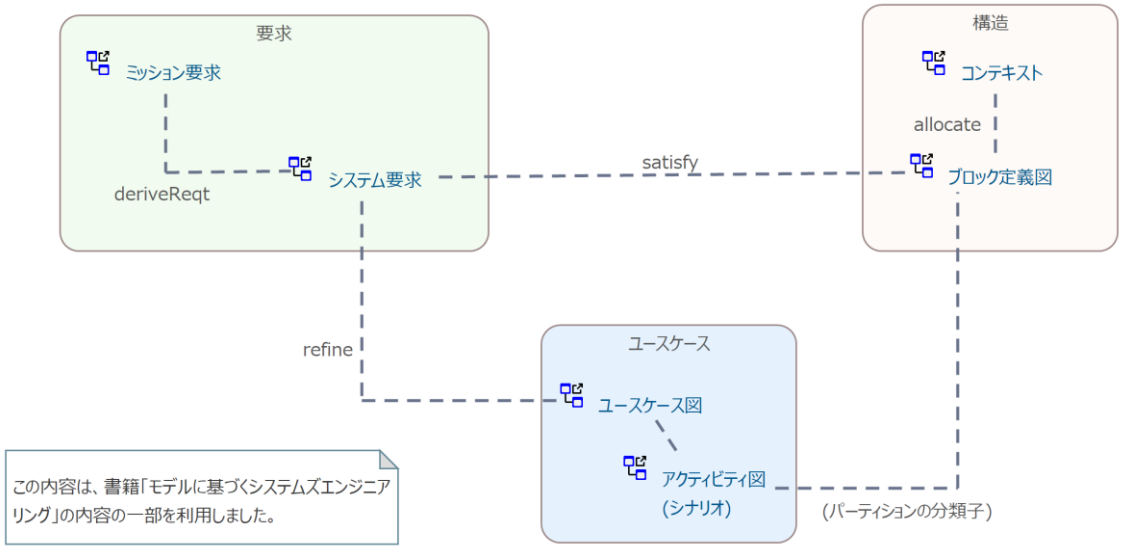
Enterprise Architect では、UML や SysML を利用した作図が可能ですが、作図をするだけであれば一般的な作図ツールでも似たような内容の図を作成できます。モデリングツールを利用することの価値として、作図ツールでは不可能なことがモデリングツールを利用すれば実現できると、それがモデリングツールを利用することの価値になります。

モデリングツールを利用すれば実現できることとして、さまざまな内容が考えられます。その中でも効果を実感しやすく、また設計の現場で求められているのが設計情報のトレーサビリティと考えます。

このドキュメントでは、トレーサビリティを中心に、作図ツールでは得られない価値の具体例を紹介します。紹介する機能の具体的な操作方法については説明していませんので、不明点がある場合にはヘルプなどの情報をご覧いただくか、サポート窓口にお問い合わせください。

システムやソフトウェアの設計では、さまざまな観点・目的で、さまざまな種類の図を作成します。このドキュメントでは SysML のいくつかの図 (ダイアグラム) を利用しシステムの要求定義からブロックへの機能の割り当てまでの内容を例として、トレーサビリティの定義と参照を紹介します。このドキュメントで説明している内容は、UML やその他の記法を利用する場合でも活用できます。

今回の例に含まれる図には、以下のような関係があります。なお、この内容の作成にあたっては、書籍「モデルに基づくシステムズエンジニアリング」で提示されているサンプルモデルの内容を参考にしています。この書籍は、特に MBSE や SysML をこれから利用する方にとって、さまざまなヒントが得られるかと思います。

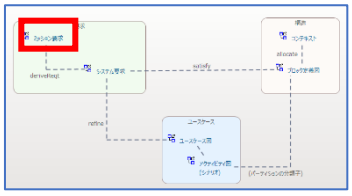


以下、それぞれの内容の説明と、関係する Enterprise Architect の機能を紹介します。実際には、それぞれの図の内容の作成を 1 回の作業で完成させることはできませんので、必要に応じて前の図に戻り、内容の追加・更新などを繰り返す必要があります。トレーサビリティに関連する機能を利用することで、内容の追加時に関係する内容への反映漏れがないか、更新時に他の部分も同時に更新する必要があるか、などを確認できます。

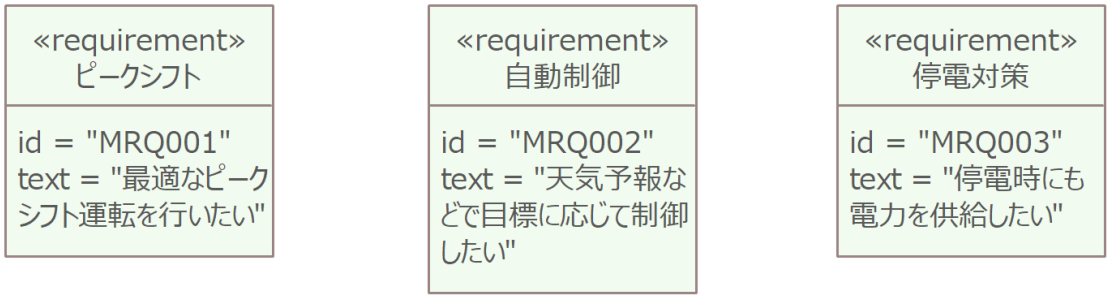
このドキュメントで利用している拡張マトリックスアドインとトレーサビリティマップアドインは、以下の URL からダウンロード・インストールが必要です。

https://www.sparxsystems.jp/products/EA/tech/Addins.htm

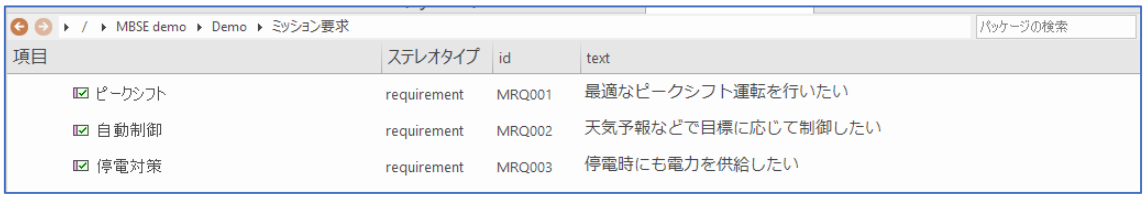
1.ミッション要求の特定



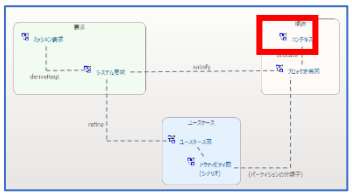
まず、どのようなシステムを作るかを検討するために、システムに求められるミッション要求を特定します。この要求は対象システムのステークホルダーによって定義されます。ここでは、SysML の要求図を用いて、SysML の要求要素として定義します。そして、ID や説明を追加します。次の図がその例です。



作成した内容を一覧として参照・編集できる、Enterprise Architect の仕様ビューも有用です。

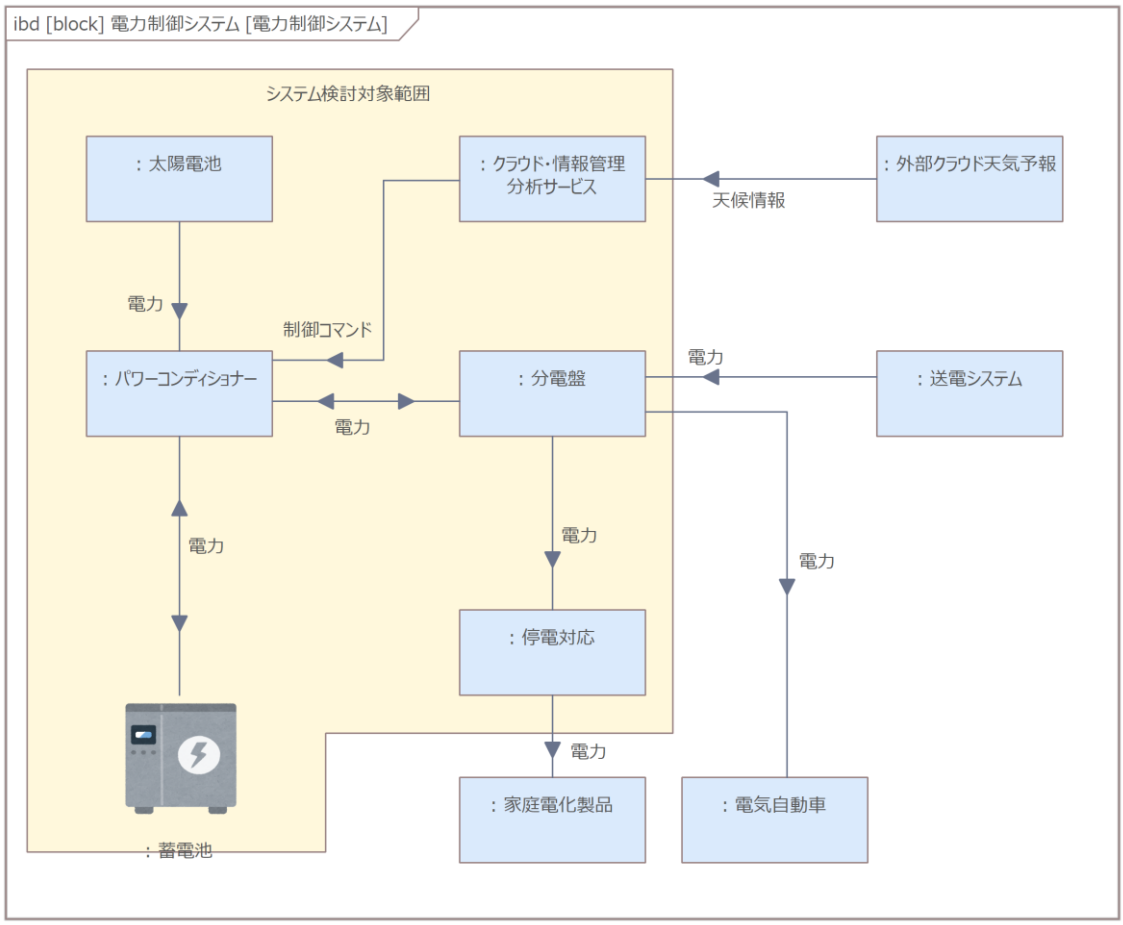


2.システムのコンテキストの明確化

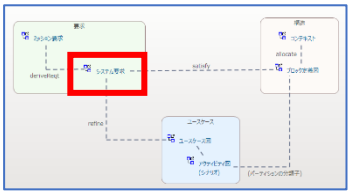


次に、システムの範囲を明確にするための「コンテキスト図」を作成します。コンテキスト図を作成することで、何を作り、何を作らないかを明確化できます。対象となるシステム以外の他のシステム・装置・サービスなど(外部要素)は設計の範囲外となり、詳細に検討する必要はありません。SysML ではコンテキスト図という図の種類はありませんので、内部ブロック図を利用することが多いです。

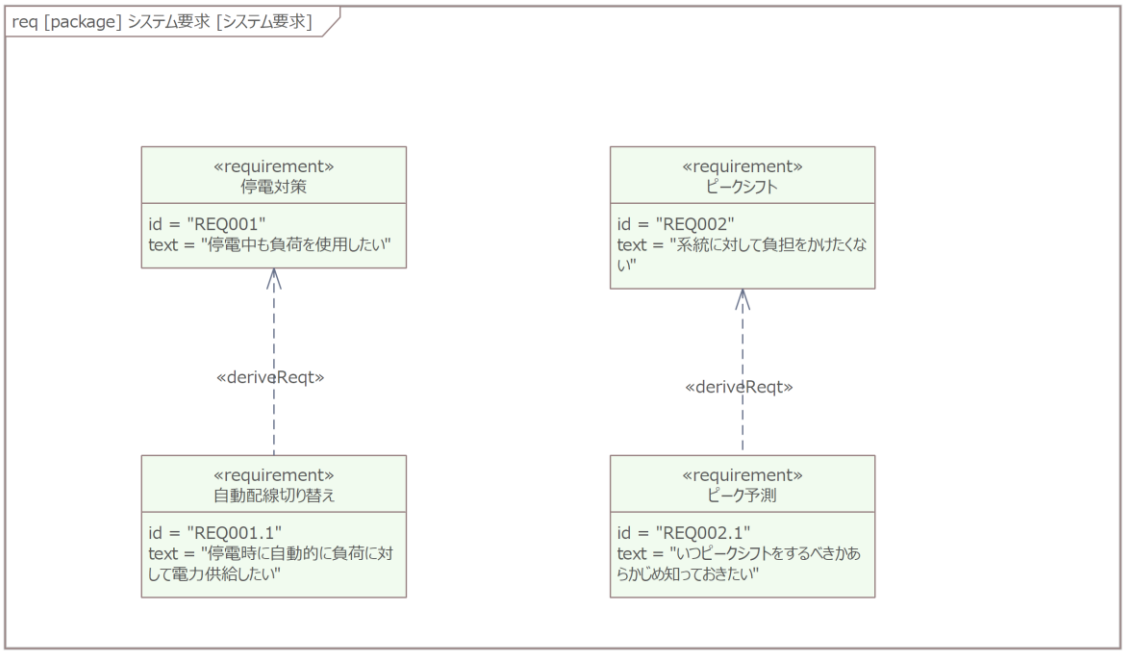
コンテキスト図では、対象のシステムと外部要素間の入出力の情報を明示することで、内容を理解しやすくなります。また、ダイアグラム内の要素に画像を適用することも、内容の理解のしやすさに影響します。



3.システム要求の特定

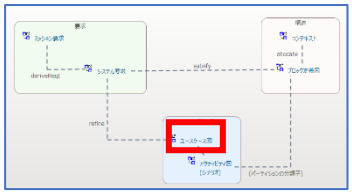


次に、対象のシステムに求められる要求を特定します。システム要求は、すべてのミッション要求を満たす必要があります。この2種類の要求の違いの一例として、ミッション要求は「ユーザーがシステムを使って何かをしたい」などユーザーの観点で表記し、システム要求は「システムが何かをしなければならない（すべき）」などシステムの観点(責務)で表記することがあります。なお、多くの場合にはシステム要求は検証可能な内容で記述されますが、今回の例ではその点までは考できていません。システム要求には、非機能要求や制約も含まれます。システム要求を複数の要求に分割して明確化する場合もあります。



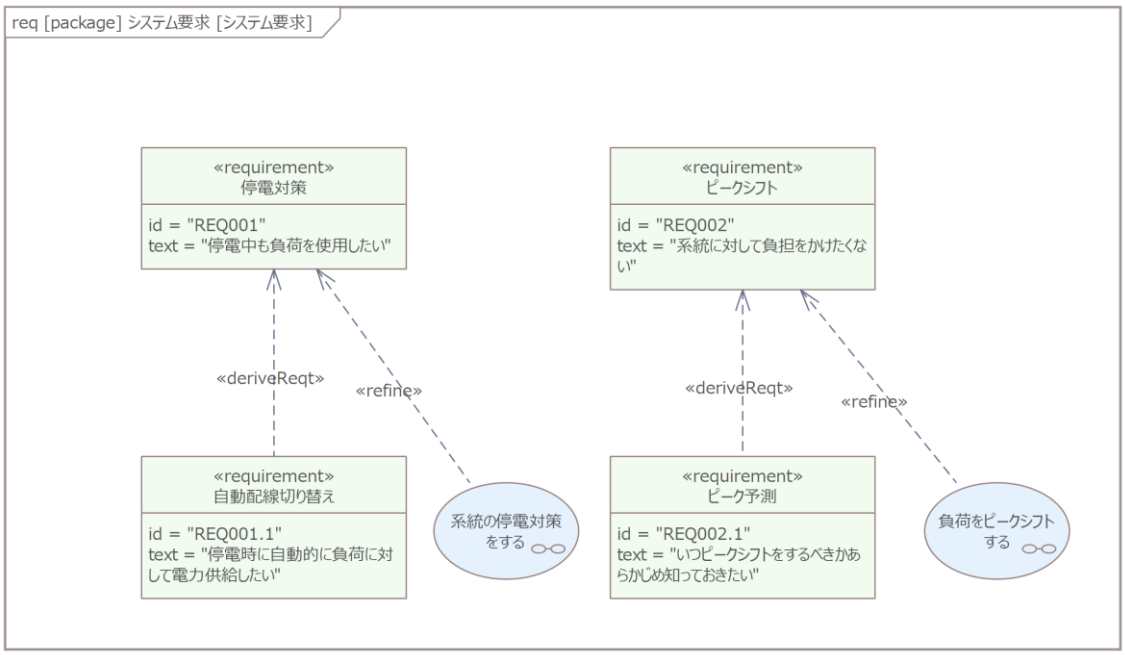
SysML の派生 (derive) や包含の接続で、ミッション要求とシステム要求の間の関係や、システム要求間の分割関係を定義します。この作業は、拡張マトリックスを使用することで効率よく追加できます。なお、図で要求間に接続を追加した場合には自動的にマトリックスのセルでは矢印として表示され、逆にマトリックスで関係を追加すると、両端の要素が同じ図に配置されている場合には要素間に接続が自動的に表示されます。

4.ユースケースの把握

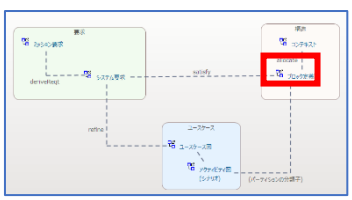


次に、ユーザーや外部要素が対象のシステムをどのように使用するかを明らかにするために、ユースケースを明確にします。通常、ユースケースの起点となるのは対象システムのユーザーになりますが、外部のシステムなど外部要素がユースケースの起点となることもあります。

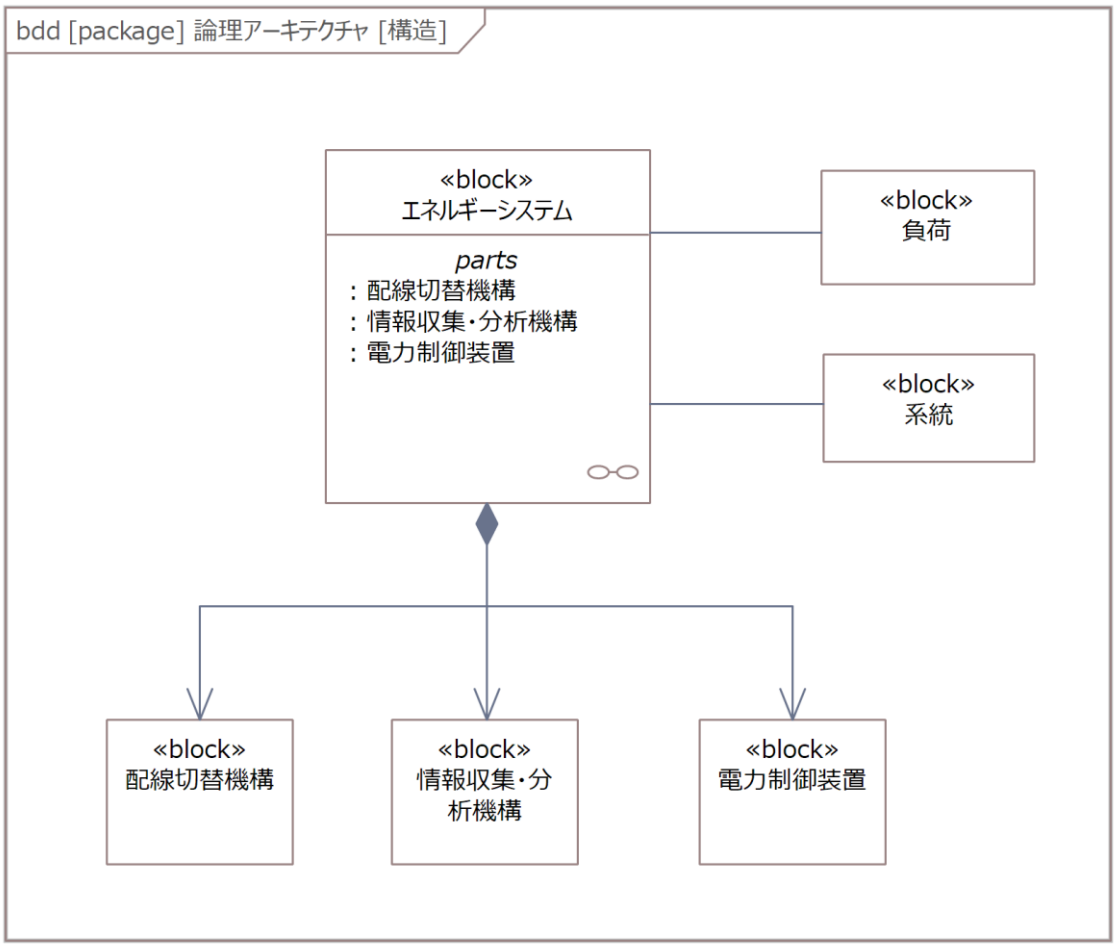
各ユースケースを、1 つまたは複数のシステム要求と結びつけています。この例では、SysML の洗練 (refine) の接続を使って、これらの間を関係づけています。図内で接続する方法の他に、拡張マトリックスを利用して結びつけることもできます。



5. 論理ブロックの定義

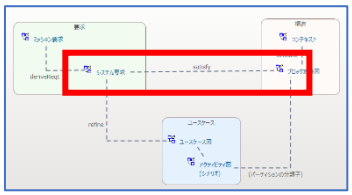


次に、論理的なブロックを SysML のブロック要素で定義し、対象システムの構造を明確化します。



必要なブロックがすべて作成されているかどうかは、後述する作業で確認できます、

6. システム要求とブロックの整合性の確認

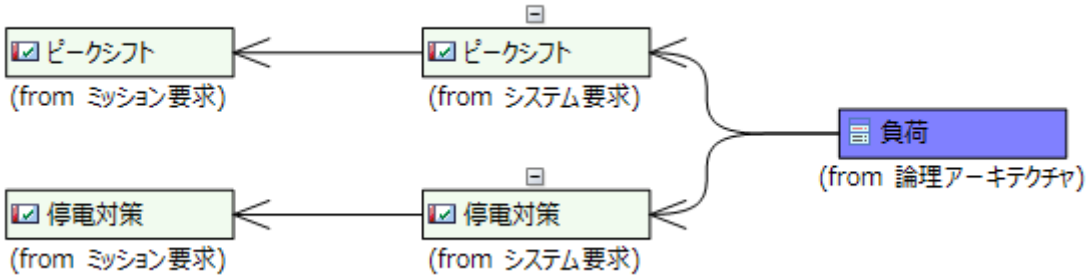


論理ブロックを定義した後、システム要求とブロックの関係を拡張マトリックスで確認します。それぞれの要求に対して、その内容を満たす (実現する) 1 つ以上のブロックが必要です。ただし、既存システムの拡張や修正を目的とする場合など、一部のブロックが要求に対応づけられていないとしても問題ない場合もあります。

ソース	論理アーキテクチャ	エネルギーシステム	系統	負荷	情報収集・分析機構	電力制御装置	配線切替機構
システム要求	7	3	2	2	0	0	0
REQ001 :停電対策	3	←	←	←			
REQ001.1 :自動配線切り替え	0						
REQ002 :ピークシフト	3	←	←	←			
REQ002.1 :ピーク予測	1	←					

(拡張マトリックス内では、要求の ID も表示しています。)

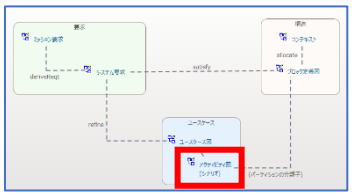
作成した関係は、トレーサビリティマップを利用することで、異なるビューで確認できます。



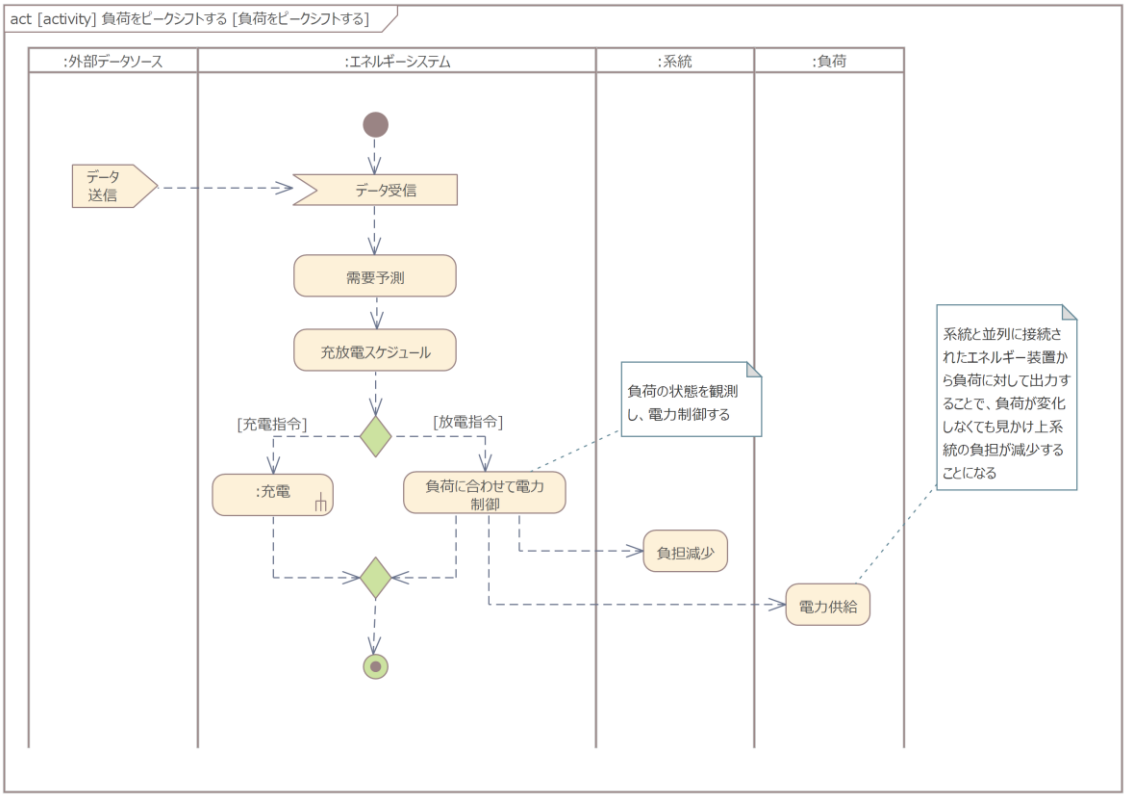
拡張マトリックスの条件を変更することで、最上位であるミッション要求と論理ブロックの間の関係をマトリックスで確認することもできます。ミッション要求とブロック要素の間には直接の接続はなく、間にシステム要求が含まれた形で間接的な関係がある状況ですが、拡張マトリックスの修正では、この間のシステム要求をマトリックスに表示せずに間接的な関係の有無の確認もできます。このような、直接接続していない要素間のマトリックスを見ることで、論理的な不整合を発見できることもあります。(あるミッション要求が、全く関係ないと思われるブロックと結びついている、等)

ターゲット	論理アーキテクチャ						
ソース	論理アーキテクチャ	エネルギーシステム	系統	負荷	情報収集・分析機構	電力制御装置	配線切替機構
ミッション要求	6	2	2	2	0	0	0
MRQ001 :ピークシフト	3	✓	✓	✓			
MRQ002 :自動制御	0						
MRQ003 :停電対策	3	✓	✓	✓			

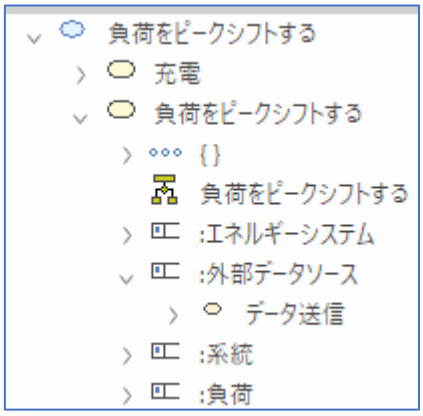
7. ユースケースの内容の明確化



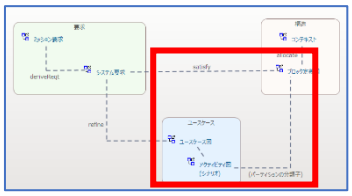
次に、それぞれのユースケースの内容を明確にするために、アクティビティ図を利用して各ユースエースのシナリオを記述します。それぞれのアクティビティ図には、関係する論理ブロックや外部システムをパーティションとして配置します。そして、処理を示すアクションを配置します。アクティビティ図には処理とデータの両方の流れを記述することが多いですが、今回の例では処理の流れのみを記述しています。 アクティビティサポートアドインを利用することで、パーティションの配置やアクションの挿入や削除などを効率よく行うことができます。



なお、ユースケース・パーティション・アクションの間には接続を追加する必要はありません。 拡張マトリックスやトレーサビリティマップでは、親子関係など接続以外の関係も対象にできます。今回の例では、以下のようにモデルブラウザの階層を構築しています。トレーサビリティマップでは、このモデルブラウザ内の親子関係も追跡の対象にできます。また拡張マトリックスでは、パーティションに配置されている要素と、そのパーティションの型となる要素間の関係を表示できます。(次の章のマトリックスをご覧ください。)



8. アクションとブロックの整合性の確認



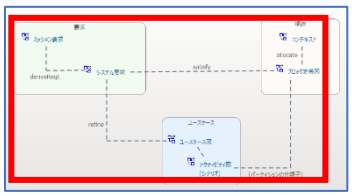
すべてのユースケースの詳細を記述した後、アクションをブロックに正しく割り当てているかを確認することで、各ブロックの責務が明確になると同時に、作成した内容に問題があれば発見の手助けになります。パーティション上のアクションは、そのパーティションに結びつくブロックの機能になります。拡張マトリックスを利用することで、全てのユースケースのシナリオにおけるブロックとアクションの関係を一目で確認できます。

このマトリックスで、以下のような確認を効率よく行うことができます。

- 同一または類似のアクションが、異なるブロックに割り当てられていないかどうか
- ブロックをいくつかの小規模なブロックに分割できないかどうか (他のブロックと比較して割り当てられているアクションが多いブロックは、ブロックの粒度が大きい可能性があります。)
- ブロックにアクションが割り当てられていない場合には、ユースケースを見逃していないか。あるいは、その上位のシステム要求やミッション要求を見逃していないか

確認の結果、必要に応じて既存のブロック・ユースケース・要求・アクションをより適切なモデルになるように修正・更新します。この作業を繰り返すことで、それぞれの図で定義した内容が洗練され、漏れや矛盾を解消できます。

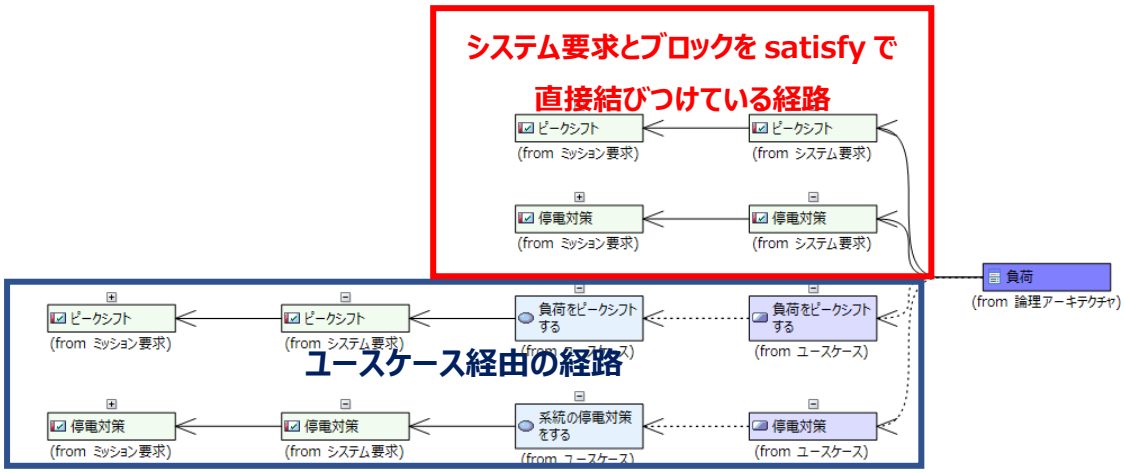
9. 矛盾・欠落の発見



最後に、トレーサビリティに関する機能を利用して、モデル全体の整合性を確認し、モデルの内容に矛盾や欠落がないかどうかを確認します。今回の例では、ミッション要求からブロックまで到達するできる 2 つの経路があります。

多くの場合には、モデルを構成する要素間のトレーサビリティの経路は 1 つのみです。この場合にはトレーサビリティ情報を漏れ・抜けの確認には利用できますが、論理的な矛盾の検出はできません。

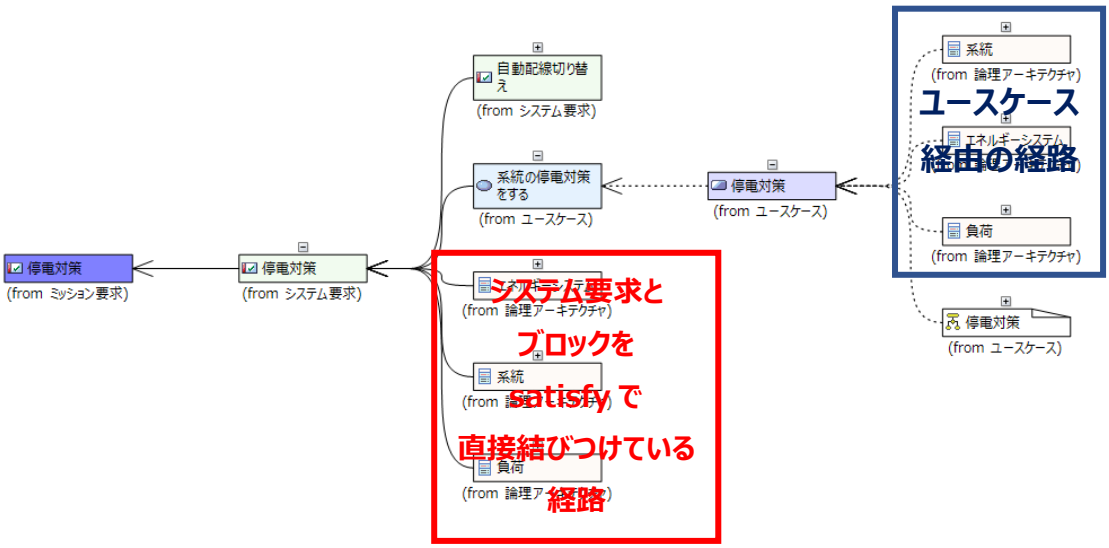
今回の例では、要求とブロックの間で、途中の要素や図が異なる 2 つの経路があります。この 2 つの経路をたどり末端が同じ要素になっているかを確認することで、モデル全体としての整合性を確認できます。例えば、次のトレーサビリティマップでは、この 2 つの経路を確認することで関係性を確認しています。下の例では、出発点は論理ブロックです。



上の例では、どちらの経路も同じミッション要求に到達しますので、内容が整合して

いることが確認できます。

同じように、ミッション要求から整合性を確認することもできます。どちらの経路でも、含まれるブロック要素の内容が一致していますので、内容が整合していることが確認できます。



こうした 2 つの経路を比較した場合に、両方の経路の末端となる要素の構成が異なる場合には、途中の内容に漏れや誤りがあることが推測されます。モデルとは、1 つの対象のシステムやソフトウェアを異なる観点で記述したものですから、その内容は全体として一貫しているはずです。

なお、こうした複数の経路での差異の確認は、今回の例ではトレーサビリティマップを展開して視覚的に確認しています。実際には、機械的・自動的に確認し問題を検出する仕組みを構築する方がよいでしょう。ただし、Enterprise Architect の機能として検出の機能はありません。その理由として、どのようなモデル・どのようなトレーサビリティ関係があるかは、それぞれの現場のモデルの構築方法に依存するため、汎用的な機能の提供が困難であることが挙げられます。こうした、それぞれの現場 (の作成物) に依存する機能は、API やアドインの仕組みを利用し、それぞれのお客様自身で構築することになります。

(弊社の有償サポートサービスで、作成を実施することも可能です。
https://www.sparxsystems.jp/advanced.htm)

まとめ

このドキュメントでは、SysML で記述されたサンプルモデルを例として、トレーサビリティの考え方などを利用してモデルの整合性や漏れ・矛盾を確認する方法を紹介しました。このように、トレーサビリティに関する機能を利用することですべてのミッション要求とシステム要求の発見を支援したり、各論理ブロックに割り当てられるべき機能(責務)の発見を支援したりできます。

また、トレーサビリティの経路を 1 つではなく複数にすることができれば、複数の経路をたどった結果から、漏れや論理的な矛盾の発見を手助けできます。

このサンプルモデルでは、論理ブロックの導出までを扱いましたが、その先の工程となる物理ブロックの導出、あるいはサブシステム・コンポーネントへの詳細化などの工程でも、トレーサビリティに関する機能をはじめとした Enterprise Architect の機能は役立ちます。

ぜひツールの機能を活用することで、作図ツールでは実現し得ない、整合性のあるモデルの構築を実現してください。