

SEMI E102-0600 (Reapproved 0706)

CIM フレームワークマテリアル搬送・格納コンポーネントに関する暫定仕様

免責事項: この SEMI スタンダードは、投票により作成された英語版が正式なものであり、日本語版は日本の利用者各位の便宜のために作成したものです。万が一英語と日本語とに差異がある場合には英語版記載内容が優先されます。

本暫定使用は、global Information & Control Committee で技術的に承認されている。現版は global Audits and Reviews Subcommittee で 2006 年 5 月 16 日に発行が承認された。2006 年 6 月に www.semi.org で、そして 2006 年 7 月に CD-ROM で入手可能となる。初版は 2000 年 6 月発行。

1 目的

1.1 半導体製造工場では、ストック格納および工程間搬送に高い基本性能が必要である。この基本性能に加えて、300 mm ウェーハキャリアのサイズが大きくなったことおよびその重量が増大したことで必要となった人間工学的および安全要求事項を考慮すると、工程内搬送が必要となる。製造装置へ、また製造装置からの材料搬送の完全自動化を経済的に実現するため、これらストック、工程間搬送および工程内搬送システムは、お互いに、そしてまた、その工場の“製造管理システム(MES)”と完全に統合する必要がある。

1.2 自動マテリアルハンドリングシステム(AMHS)装置の基本的要求事項は、工場 MES との効率的な統合である。そのため、本スタンダードの目的は、図 1 に示しているとおり、相互運用性を持つ AMHS システムを経済的に統合できるようにすることである。メーカーは、AMHS 統合システムの目に見える動作および工場 MES とのインタフェースを規定する SEMI スタンダードを求めている。本スタンダードの目的は、これらインタフェースおよび CIM フレームワークの一部として MES と AMHS システム間の連係を規定することである。AMHS という言葉は自動マテリアルハンドリングをサポートする装置およびシステムを幅広く指すものとして一般的に使われているが、CIM フレームワークは、AMHS 性能一揃いに対する MES のスタンダードインタフェースを表すマテリアル搬送・格納コンポーネント(MTSC)と呼ぶソフトウェアコンポーネントを規定する。

2 範囲

2.1 本スタンダードは、(他の CIM フレームワークコンポーネントで代表される)工場とマテリアルトラッキング・格納コンポーネント間での実行時の相互連係に対して、製造管理システム(MES)が必要とする共通インタフェースを提供する。マテリアル搬送および格納装置の構造とトラッキングを支援するインタフェースの一部は、CIM フレームワークの装置トラッキング・メンテナンスコンポーネントが担当する。それらインタフェースは、本スタンダードに記述した搬送マシンおよび格納マシンのインタフェースを補間する。

2.2 マテリアル搬送・格納コンポーネントの範囲には、下記のインタフェースが含まれる。

- 特定場所へのマテリアル搬送に要する時間を予測して、マテリアル搬送のスケジューリングおよび処理を支援する。注: 配送時間予測のためのインタフェースは、本暫定スタンダードに注記したように、まだ決定していない部分として今後の活動を必要としている。
- マテリアルを特定の場所に移動する搬送ジョブを実行し、監視する。
- ジョブが実行可能かどうか検証する: マテリアルは存在するか、目的地に到達可能か、格納あるいはロードポート容量が使用可能かおよびジョブを実行するのに必要なマテリアル移動リソース(車、保管スペースなど)は十分であるか。
- 内部マテリアル移動および格納を実行するため、AMHS 装置コントローラに作動要求を行う。マテリアル搬送・格納コンポーネントを実行するには、物理的移動および格納動作を行う装置との間の通信に IBSEM およびストック SEM などの低レベルのスタンダードを使用する。
- 搬送ジョブの実行および履歴についてのデータを収集し、記録する。
- 装置のイベントおよび故障検出を捕捉、記録、解釈および応答をする。

- MESレベルのジョブステータスイベントおよびマテリアル場所変更イベントを発生させる。
- システム中にあるマテリアルについてその所在位置およびその搬送履歴を記録し報告する。
- マテリアルのハンドオフ、ハンドシェイクプロトコルに対するマシンおよびポートインタフェースと連携する。

2.3 自動マテリアルハンドリングシステム(AMHS)は、どの半導体工場でも重要な部分であり、通常実施されていて、別個の論理ソフトウェア要素として製造管理システム(MES)と統合されている。AMHS は、AMHS フレームワーク、AMHS 装置のためのシステムコントローラ(ソフトウェア)および搬送、格納マシン(ハードウェア)で構成されている。この伝統に従って、CIM フレームワークは、AMHS を「ブラックボックス」とみなす。このシステムへのインタフェースは目に見えるが、そのシステム内部の動きやコントロールは AMHS サプライヤの責任であって、MES の領域ではない。CIM フレームワークでは、マテリアル搬送・格納コンポーネント(MTSC)を MES レベルでの AMHS 機能とのインタフェースと規定している。

注意: このスタンダードはその使用に関連したすべての安全問題を取り扱うことを意図していない。このスタンダードの使用者は、その責任において、適切な安全および健康上実施すべき事柄を確立し、また使用前に法規制やその他の制限への適用性を判断するものである。

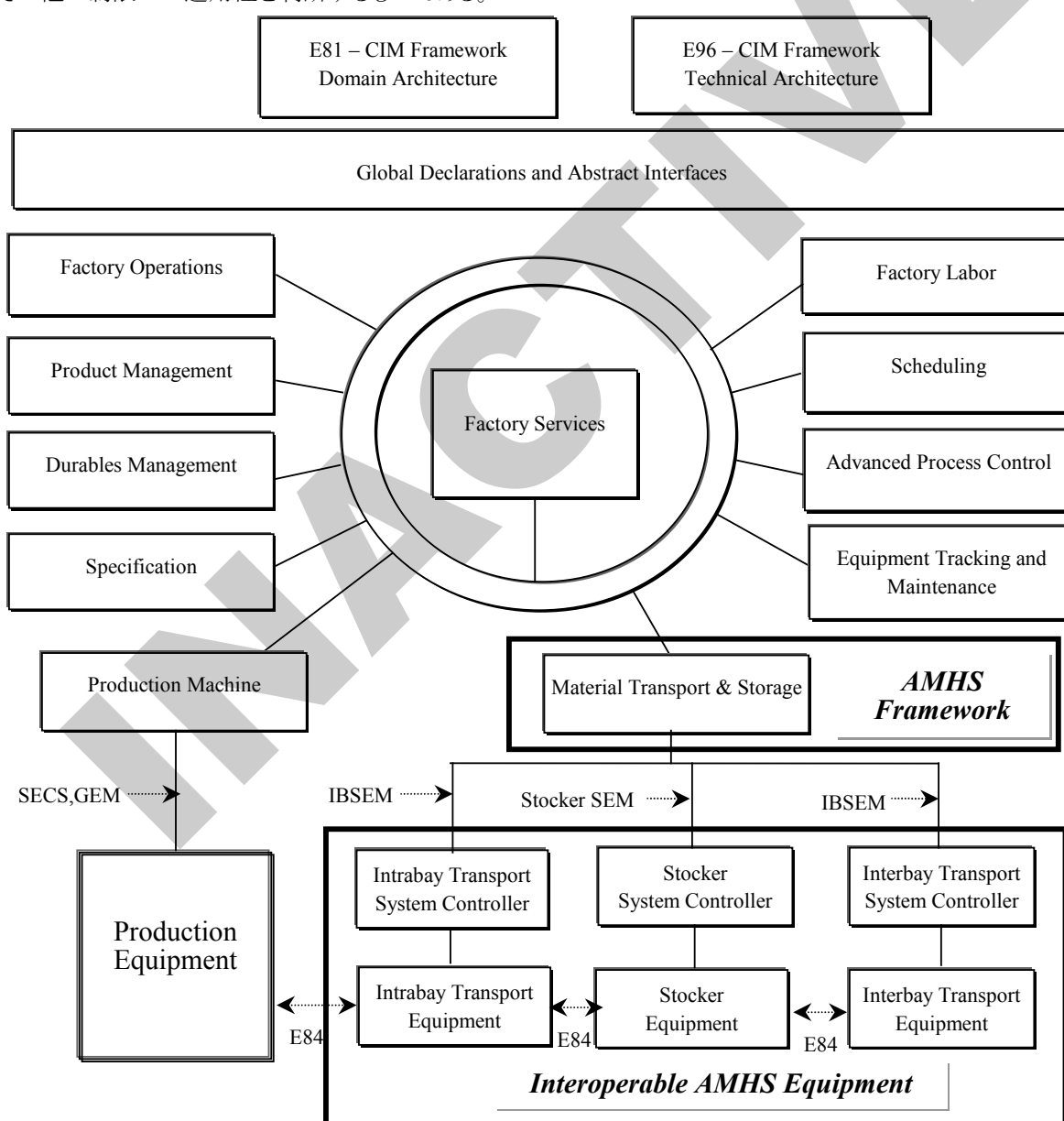


図 1
AMHS 環境

3 制限

3.1 マテリアル搬送・格納コンポーネントは、相互運用を行う AMHS 装置に AMHS フレームワークレベルのインタフェースを提供する。このレベルのマテリアル搬送は、工場レベルの移動、つまり、マシン間、エリア間あるいはマテリアルトラッキングロケーション間で、製品および耐久材を搬送することである。このマテリアルトラッキング・格納コンポーネントは、マシン内でマテリアルが移動するのを目に見えるようにするインタフェースを提供するものではない。

3.2 暫定ステータス — 本仕様は未完成な部分があるため、暫定とされている。今後、この文書をフルスタンダードにする前に解決しなければならない点は下記の通りである。

3.2.1 グローバル宣言および抽象インタフェースへの依存 — この文書の構成部分には、グローバル宣言、マテリアルに関する特化した抽象インタフェース、ならびにリソースおよびジョブコンセプトを参照しているものがある。1999 年 7 月に SEMI Information and Control Committee が、この宣言およびインタフェースに対する暫定スタンダードを提案した。このスタンダードが発行される際には、CIM フレームワークの共通エレメントと本仕様との関係を明確に定義するための改訂作業に着手することになる。

3.2.2 耐久材マネジメントコンポーネントへの依存 — マテリアル搬送・格納コンポーネントは、CIM フレームワークの耐久材マネジメントコンポーネントの一部として規定されるインタフェースに依存する。本文書には耐久材マネジメントコンポーネントが必要とするインタフェースだけを含んでいる。そのインタフェースは § 6.2.3 に掲載しているが、CIM フレームワークの耐久材マネジメントコンポーネントが将来、投票で採用されたときには、この文書からは削除されることになる。

3.2.3 マシン抽象インタフェースグループへの依存 — マテリアル搬送・格納コンポーネントは、CIM フレームワークのマシン抽象インタフェースグループ(マシン AIG)の一部として規定されているインタフェースに依存している。本文書にはマシン AIG からは必要なインタフェースしか記されていない。そのインタフェースは § 6.2.4 に掲載されているが、CIM フレームワークのマシン AIG のスタンダードが将来成立したら本文書からは削除される。

3.2.4 ファクトリオペレーションコンポーネントへの依存 — マテリアル搬送・格納コンポーネントは、CIM フレームワークのファクトリオペレーションコンポーネントの一部として規定されているインタフェースに依存している。本文書にはファクトリオペレーションコンポーネントからは必要なインタフェースしか記されていない。そのインタフェースは § 6.2.5 に掲載されているが、CIM フレームワークのファクトリオペレーションコンポーネントのスタンダードが将来成立したら本文書からは削除される。

3.2.5 搬送時間の決定性予測 — 将来の TransportJobs の予定搬送時間を評価するという作業は暫定仕様の範囲を超えている。この性能はそういった予測に関する経験を積んだ上で、それに基づいて、将来本仕様で正式スタンダードとして改訂される際に追加されるかもしれない。搬送時間の予測には、発見的方法が最も現実的な方法であるということがわかるかもしれない。

4 参照スタンダードおよび文書

4.1 SEMI スタンダード

SEMI E81 — Provisional Specification for CIM Framework Domain Architecture

4.2 その他の文書

ISO/IEC International Standard 14750 (also ITU-T Recommendation X.920): Information Technology — Open Distributed Processing — Interface Definition Language¹

UML Notation Guide, Version 1.1, document number ad/97-08-05, Object Management Group²

¹ ISO Central Secretariat, 1, rue de Varembé, Case postale 56, CH-1211 Genève 20, Switzerland
(国内連絡先) 日本規格協会 〒107-0052 東京都港区赤坂 4-1-24 03-3583-8003

5 用語

5.1 略語および頭字語

5.1.1 *AMHS* — 自動マテリアルハンドリングシステム (Automated Material Handling System)

5.1.2 *MTSC* — マテリアル搬送・格納コンポーネント (Material Transport and Storage Component)

6 要求条件

6.1 本項には CIM フレームワークのマテリアル搬送・格納コンポーネントを構成するインタフェースが規定されている。さらに、マテリアル搬送・格納をサポートするため必要となる耐久材マネジメント、マシン抽象インタフェースグループおよびファクトリオペレーションとのインタフェースのサブセットもここに含まれているが、これらの仕様は、今後の投票にかけられるものである。

6.1.1 図 2 に、CIM フレームワークに含まれるマテリアル搬送・格納コンポーネントの内容を図示している。ファクトリオペレーションコンポーネントは、搬送ジョブを生成するマテリアル搬送マネージャにマテリアル搬送を行うことを要求する。マテリアル搬送・格納コンポーネントは、マシン AIG が提供するサービスを使って、搬送ジョブの出発点または目的地となるマシンおよびポートのインタフェースにアクセスする。MES ファクトリはマシンのグループを表す工場内の領域を定義する。最後に、耐久材マネジメントは移動するマテリアル容器の管理に必要なサービスを提供する。

6.1.2 図 3 は継承されたインタフェースならびにインタフェースの相互関連を含んだ、マテリアル搬送・格納コンポーネントのインフォメーションモデルを示す。これらのインタフェースは § 6.3 および § 6.4 に詳細に定義されている。マテリアル搬送・格納に関連するインタフェースは次の通りである。

1. TransportJobSupervisor — 抽象インタフェース
2. MaterialTransportManager
3. MaterialTransportController
4. TransportJob

6.2 将来の CIM フレームワーク仕様に必要なサブセット

6.2.1 この仕様は、耐久材マネジメントコンポーネント、マシン抽象インタフェースグループおよびファクトリオペレーションコンポーネントのインタフェースに依存しており、それらは最終的に将来の CIM フレームワークコンポーネントスタンダードに含まれることになる。それらはこのスタンダードに含まれていて、CIM フレームワーク内の AMHS サポートに関する仕様を完全なものにしている。それらが投票によって別個のスタンダードとなれば、このスタンダードから削除されて、それら自身で完全なスタンダードとなって参照されることになる。

6.2.2 図 4 は、マテリアル搬送・格納コンポーネントの依存性を示す。

2 UML Notation Guide v1.1 is available to the general public at <http://www.omg.org/cgi-bin/doclist.pl>, +1-508-820 4300, Object Management Group, Inc., Framingham Corporate Center, 492 Old Connecticut Path, Framingham, MA 01701

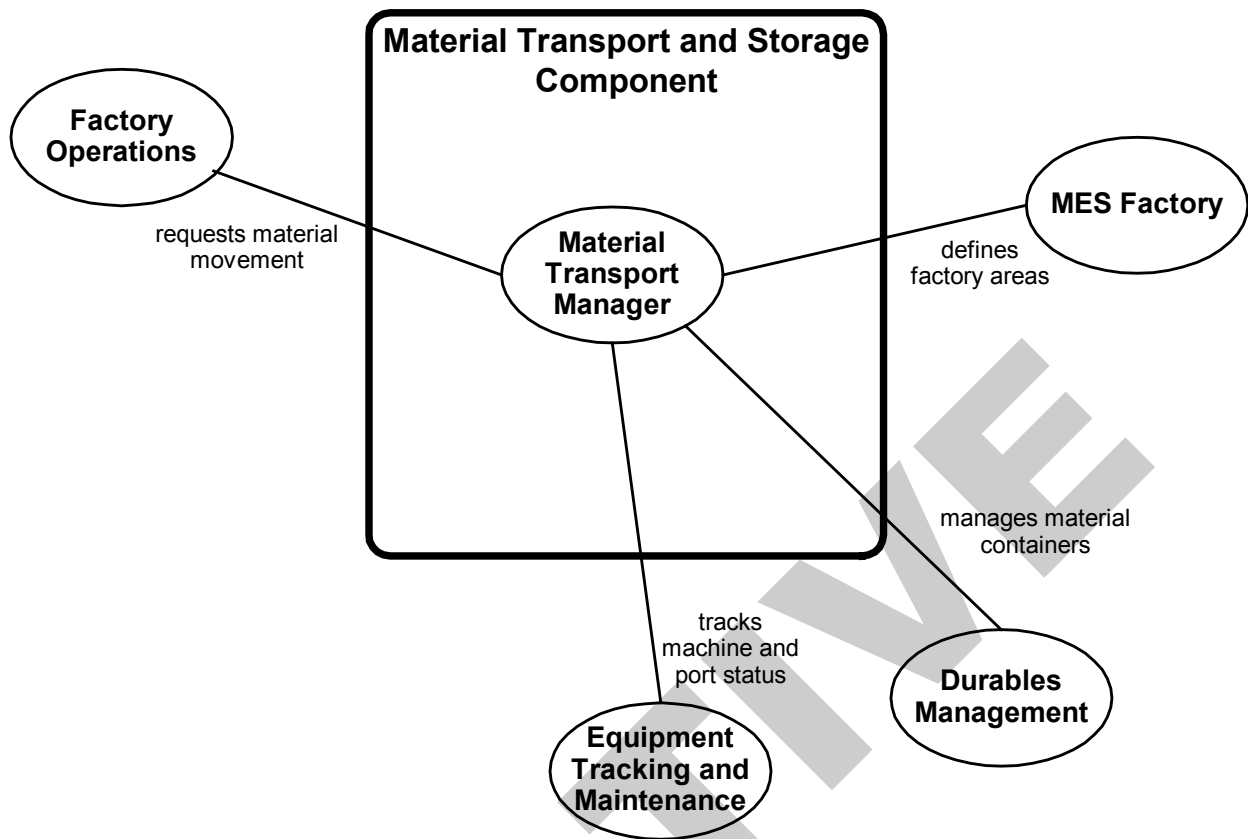


図 2
マテリアル搬送・格納コンポーネントのコンテキスト

Material Transport and Storage Component

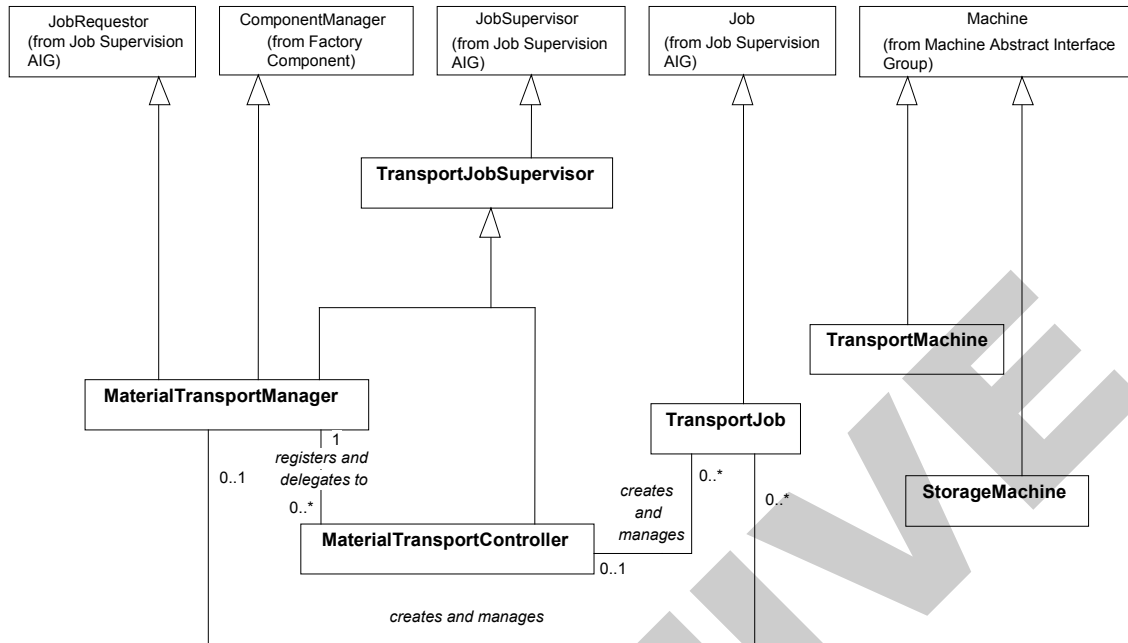


図 3
マテリアル搬送・格納の情報モデル

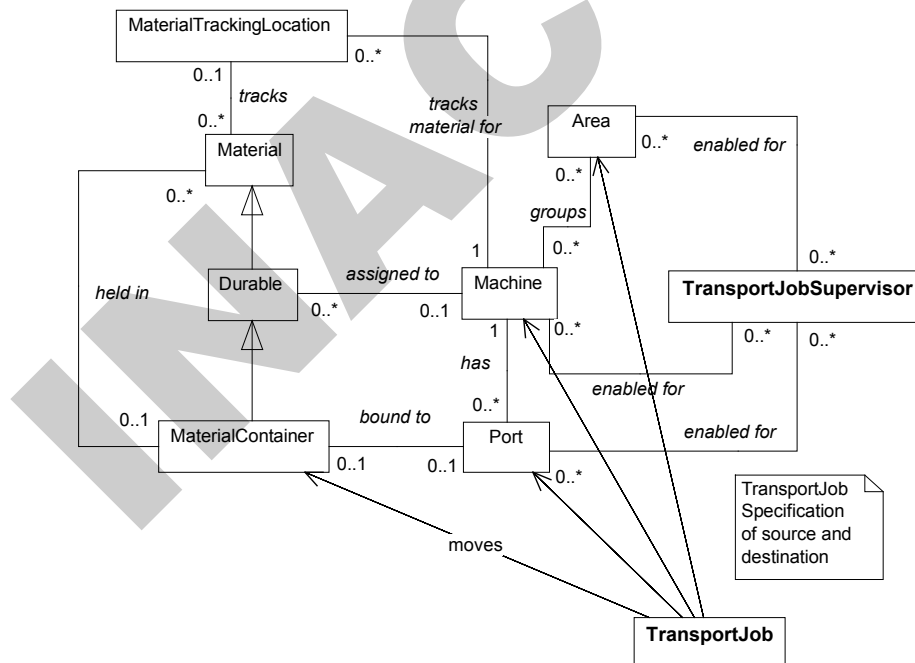


図 4
マテリアル搬送・格納の依存性

6.2.3 耐久材マネジメントコンポーネント(サブセット) – この節には、このコンポーネントスタンダードで引用した耐久材マネジメントコンポーネントからのインタフェースおよび特定のオペレーションを含んでいる。それらインタフェースは、このスタンダードの引用上の整合性を図るためここに記載されているのであって、このコンポーネントの一部として記載されているのではない。さらに、これは必要なインタフェースを包括的に扱うことを意図するためではなく、耐久材マネジメントの仕様全体が次の投票で採用される前にそのサブセットだけをここに参照用として提示してあるにすぎない。耐久材マネジメントのスタンダードが成立すると、このスタンダードはその後の投票で改訂され、そのインタフェースをここに載せず、直接参照するようになる。

6.2.3.1 耐久材インタフェース(サブセット)

モジュール: DurablesManagement

インタフェース: Durable

継承インタフェース: Material

```
interface Durable : AbstractIF::Material {
```

説明: 耐久材インタフェースは、製造を促進するために使用する工場内の資材を指し、通常プロセスで消費されることはない。それは工場内で場所を移動することが可能で、ダイナミックなトラッキングを必要とする。これには、マテリアルを搬送する容器、固定器具(マテリアルを一定の位置に保持しておくための付属品)、および製造プロセスで装置または従業員が使用する道具(レチクル、ロードボード、ワークホルダなど)を含む。耐久材(指定プロセスで使用可能なすべてのレチクルなどの)をグループに分けるには、必要な特定カテゴリに対するMaterialGroupsを生成すればよい。

例外:

公開イベント:

/*耐久材をトラッキングする MaterialTrackingLocation の変更を利用者に通知する。*/

DurableLocationChangedEvent

提供サービス:

/*耐久材の場所を設定し、入手する。MaterialTrackingLocation が NULL にセットされている場合、その耐久材はどの MaterialTrackingLocation にも存在していなくて、工場内のマニュアルコントロールのもとにあることがある。*/

```
void setMaterialTrackingLocation (
    in MachineAIG::MaterialTrackingLocation
        aMaterialTrackingLocation)
    raises (Global::FrameworkErrorSignal,
        MachineAIG::MaterialTrackingLocation::MaterialTrackingLocationFullSignal);
```

```
MachineAIG::MaterialTrackingLocation getMaterialTrackingLocation ( )
    raises (Global::FrameworkErrorSignal);
```

/*ある特定マシンに対してどの MaterialTrackingLocation でこの耐久材をトラッキングするかを決めるユニットを設定し、取得する。このユニットは、MaterialTrackingLocation のユニット属性にマッピングした一連の値である。*/

```
void setUnit(
    in string aUnit)
    raises (Global::FrameworkErrorSignal,
        マシン AIG:: MaterialTrackingLocation::InvalidUnitSignal);
```

```
string getUnit( )
    raises (Global::FrameworkErrorSignal);
```

```
}; // Durable
```

動的モデル: なし

6.2.3.2 MaterialContainer インタフェース(サブセット)

モジュール: DurablesManagement

インタフェース: MaterialContainer

継承インタフェース: Durable

```
interface MaterialContainer : Durable {
```

説明: MaterialContainer インタフェースは、搬送、処理または格納のためにマテリアルを収納しておく容器を表す。MaterialContainer の例としては、出荷用ボックス、出荷用チューブ、スタンダードメカニカルインタフェース(SMIF)ボッドなどがある。

例外: なし。

公開イベント:

/* MaterialContainer に発生した状態変化を利用者に通知する。*/

MaterialContainerStateChangeEvent

提供サービス:

/* MaterialContainer 状態遷移を引き起こすため、次のオペレーションが設けられている。*/

```
void makeManualControl ( )  
    raises (Global::FrameworkErrorSignal,  
            Global::InvalidStateTransitionSignal);  
void makeStored ( )  
    raises (Global::FrameworkErrorSignal,  
            Global::InvalidStateTransitionSignal);  
void makeInTransit ( )  
    raises (Global::FrameworkErrorSignal,  
            Global::InvalidStateTransitionSignal);  
void makeProcessing ( )  
    raises (Global::FrameworkErrorSignal,  
            Global::InvalidStateTransitionSignal);  
}; // MaterialContainer  
typedef sequence <MaterialContainer> MaterialContainerSequence;
```

動的モデル:

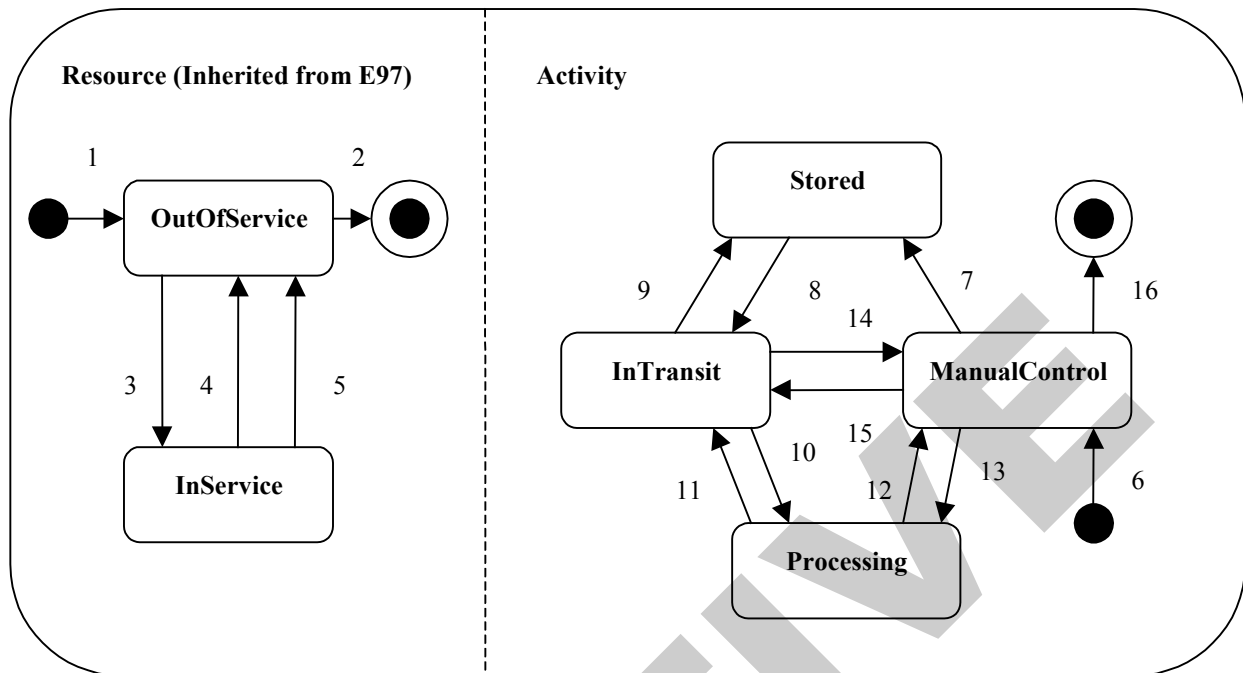


図 5
MaterialContainer の動的モデル

表 1 MaterialContainer の状態定義

状態	状態定義	例
InService	Resource 状態モデルから継承	
OutOfService	Resource 状態モデルから継承	
InTransit	コンテナは搬送ジョブの対象として搬送中	コンテナは OHT 搬送ビークルに載っている
Stored	コンテナはか格納マシンに格納されており, 搬送ジョブの対象ではない	コンテナはストックの棚にある
ManualControl	コンテナは“マテリアル搬送・格納コンポーネント”のコントロール外にある	コンテナは PGV に載っている
Processing	コンテナはプロセス装置にある	コンテナはリソグラフィツール上にある

表 2 MaterialContainer 状態遷移

#	現在の状態	トリガー	新しい状態	アクション	コメント
1~5	注: 遷移 1~5 は, リソースから継承した動的モデルに定義されている。				
6	なし	makeManualControl()	ManualControl	MaterialContainer インスタンスが生成される	初期状態
7	ManualControl	makeStored()	Stored	DurableLocationChangedEvent を MaterialContainer インスタンスが発行	コンテナは格納マシンの搭載ポートに置かれて, 搬送ジョブが割り当てられていない
8	Stored	makeInTransit()	InTransit	DurableLocationChangedEvent を MaterialContainer インスタンスが発行	コンテナは移動を開始。TransportJob 状態は Executing に遷移

#	現在の状態	トリガー	新しい状態	アクション	コメント
9	InTransit	makeStored()	Stored	DurableLocationChangedEventを MaterialContainer のインスタンスが発行	コンテナは格納。TransportJob 状態は Completed に遷移
10	InTransit	makeProcessing()	Processing	DurableLocationChangedEventを MaterialContainer のインスタンスが発行	コンテナはプロセス装置の仕向地に到着。TransportJob 状態は Completed に遷移
11	Processing	makeInTransit()	InTransit	DurableLocationChangedEventを MaterialContainer のインスタンスが発行	コンテナはプロセス装置を離脱。TransportJob 状態は Executing へ遷移
12	Processing	makeManualControl()	ManualControl	DurableLocationChangedEventを MaterialContainer のインスタンスが発行	コンテナはプロセス装置のコントロールから離脱 (TransportJob を経由しない)
13	ManualControl	makeProcessing()	Processing	DurableLocationChangedEventを MaterialContainer のインスタンスが発行	コンテナはプロセス装置のコントロール下に入る (TransportJob を経由しない)
14	InTransit	makeManualControl()	ManualControl	DurableLocationChangedEventを MaterialContainer のインスタンスが発行	コンテナはマニュアル払い出しポートを経由 TransportJob のコントロールから離脱。TransportJob 状態は Aborted へ遷移
15	ManualControl	makeInTransit()	InTransit	DurableLocationChangedEventを MaterialContainer のインスタンスが発行	コンテナは手動受け入れポートを経由 AMHS に入る。TransportJob 状態は Executing へ遷移
16	ManualControl	なし	適用なし	なし	MaterialContainer インスタンスを消去

6.2.4 マシン抽象インタフェースグループ(サブセット)

6.2.4.1 本項は、マシン抽象インタフェースグループのインタフェースおよび特定オペレーションで、本コンポーネントスタンダードに参照したものだけを扱っている。これらインタフェースは、このスタンダードの整合性を図るためここに記載されており、このコンポーネントの一部として残すことを意図していない。さらに、これは必要なインタフェースを包括的に扱うつもりはなく、マシン抽象インタフェースグループの完全なスタンダードが投票で成立するまで、そのサブセットを参照するために必要となるに過ぎない。、マシン抽象インタフェースグループのスタンダードが成立すると、このスタンダードは改訂され、そのインタフェースをここに記載するのではなく直接参照するようになる。

Error! Objects cannot be created from editing field codes.

図 6

マシン抽象インタフェースグループ（サブセット）の情報モデル

6.2.4.2 Machine インタフェース(サブセット)

モジュール: マシン AIG

インタフェース: マシン

継承インタフェース: Resource

```
interface マシン : AbstractIF::Resource {
```

説明: この抽象インタフェースは、工場内にある装置一台の概念を表す。マシンは、MES コンテキストの中を参照できるよう、工場内に存在する物理的装置の固体識別をおこなう。また、このマシンは、自身で処理し、使用しまたは格納するマテリアルを分類するために定義したユニットごとに一つの MaterialTrackingLocation を保持する。

タイプ定義:

```
typedef sequence <マシン> マシン Sequence;
```

```
typedef sequence <MaterialTrackingLocation> MaterialTrackingLocationSequence;
```

```
typedef sequence <Port> PortSequence;
```

```
enum マシン Type {
    FixedBufferProduction マシン,
    InternalBufferProduction マシン,
    Storage マシン,
    Transport マシン };
```

例外:

```
/*マテリアル受け取りのため指示されたポートは無効だった。*/
```

```
exception InvalidPortSignal { };
```

```
/*アンロード指定されたマテリアルは無効だった。*/
```

```
exception InvalidMaterialSignal { };
```

公開イベント:

```
/*E10 状態変化がマシンで発生し、これを工場側が知っておく必要がある。*/
```

```
マシン ResourceE10StateChangedEvent
```

提供サービス:

```
/*このマシンの MachineType を設定する。*/
```

```
void setMachineType (
    in MachineType aMachineType )
    raises (Global::FrameworkErrorSignal);
```

```
/*このマシンの MachineType を返す。*/
```

```
MachineType getMachineType ( )
    raises (Global::FrameworkErrorSignal);
```

/*このマシンの MaterialTrackingLocation を設定して、中身を特徴付けている特定マテリアルユニットをトラッキングする。MaterialTrackingLocation がそれぞれ別個のユニットをトラッキングする間は、マシンは一個以上の

MaterialTrackingLocation を持つことができる。ユニット値は、トラッキングされるマテリアルをマシンの正しい MaterialTrackingLocation に割り当てるため、マテリアルの対応属性と共に使用される。*/

```
void setMaterialTrackingLocation (  
    in MaterialTrackingLocation aMaterialTrackingLocation,  
    in string aUnitToTrack)  
    raises (Global::FrameworkErrorSignal);
```

/*このマシンに対するすべての MaterialTrackingLocation を入手する。マシンで使用したユニット当たり MaterialTrackingLocation はただ一つだけである。*/

```
MaterialTrackingLocationSequence getAllMaterialTrackingLocations (  
    raises (Global::FrameworkErrorSignal);
```

/*指定ユニットに対する MaterialTrackingLocation を入手する。*/

```
MaterialTrackingLocation getMaterialTrackingLocation (  
    in string aユニット)  
    raises (Global::FrameworkErrorSignal);
```

/* マシンにマテリアルをロードするポートを確保するよう要求する。これには、ポートとマシンオブジェクトの間の通信が含まれる場合があるが、公開である必要はない(またはマシンと装置間の通信が含まれる場合がある)。RequestedPort パラメータが null のままであれば、マシンが requestedPort を選択する。マシンが従うことができない場合または拒否した場合、Null を返す。*/

```
Port reservePortForTransferTo (  
    in DurablesManagement::MaterialContainer aContainer,  
    in Port requestedPort )  
    raises (Global::FrameworkErrorSignal,  
        InvalidPortSignal);
```

/* マシンに 2 個以上の MaterialContainers をロードするためのポートを確保するよう要求する。シーケンスに含まれている MaterialContainers がすべてロードされた後でないと Machine は別のポート予約要求を受諾できない。MaterialContainerSequence で指定された MaterialContainers をロードする順番は決まっていない。RequestedPort パラメータが null のままなら、マシンが requestedPort を選択する。Machine が従うことができない場合あるいは拒否した場合は Null を返す。*/

```
Port reservePortForBatchTransferTo (  
    in DurablesManagement::MaterialContainerSequence aContainerGroup,  
    in Port requestedPort )  
    raises (Global::FrameworkErrorSignal,  
        InvalidPortSignal);
```

/*あるポートへの搬送に対して行われた予約を取消すようマシンに要求する。*/

```
void cancelPortReservationForTransferTo (  
    in DurablesManagement::MaterialContainer aContainer,  
    in Port requestedPort )  
    raises (Global::FrameworkErrorSignal,  
        InvalidPortSignal);
```

/*あるポートへの搬送またはポートからの搬送が完了したことをマシンに通知する。これにはポートとマシンオブジェクト間の通信が含まれることがあるが、公開である必要はない(マシンと装置間の通信が含まれることがある)。*/

```
void transferComplete (in DurablesManagement::MaterialContainer aContainer,  
    in Port reservedPort )  
    raises (Global::FrameworkErrorSignal,  
        InvalidPortSignal);
```

/*マテリアルをアンロードする準備をするようマシンに要求する。マテリアルがアンロードされたポート、またはマシンが従うことができない場合または拒否した場合、Null を返す。*/

```
Port containerOut (
    in DurablesManagement::MaterialContainer aContainer,
    in Port requestedPort )
    raises (Global::FrameworkErrorSignal,
           InvalidMaterialSignal);

/*そのマシンに対するすべてのポートリストを返す。*/

PortSequence allPorts ( )
    raises (Global::FrameworkErrorSignal);

/*このマシンが入っているすべての Area リストを返す。*/

AreaSequence allAreas( )
    raises (Global::FrameworkErrorSignal);

}; // Machine
```

6.2.4.3 *MaterialTrackingLocation* インタフェース

モジュール: MachineAIG

インタフェース: MaterialTrackingLocation

継承インタフェース: OwnedEntity

```
interface MaterialTrackingLocation : AbstractIF::OwnedEntity {
```

説明: MaterialTrackingLocation は、Material を保持しておく場所である。ロケーションの「タイプ」は、ロケーションが保持できる Material の分類ユニットに基づく(例えば、ウェーハを保持するロケーションは「ウェーハ」ユニットを持ち、この Material タイプだけ保持できる)。各マシンはマテリアルユニットごとに一つの MaterialTrackingLocation を持っている。
MaterialTrackingLocation にも、許容収容力がある。

例外:

/*コールで指定された Material は、この MaterialTrackingLocation に対して有効な Material ではない。*/

```
exception InvalidMaterialSignal { };
```

/* MaterialTrackingLocation にさらに Material を割り当てようとしたが、満杯であるため失敗した。*/

```
exception MaterialTrackingLocationFullSignal { };
```

/* ユニット値が、MaterialTrackingLocation に対して有効ではない。*/

```
exception InvalidUnitSignal { };
```

公開イベント:

/* MaterialTrackingLocation が、その収容力一杯になった。*/

MaterialTrackingLocationFullEvent

/*前に満杯であった MaterialTrackingLocation の収容力に今、空きがある。*/

MaterialTrackingLocationNotFullEvent

提供サービス:

/* MaterialTrackingLocation にある Material を返し、ロケーションが空であれば nil を返す。*/

```
AbstractIF::MaterialSequence allMaterialHeld ( )
    raises (Global::FrameworkErrorSignal);
```

/* MaterialTrackingLocation にユニークな識別子を取得し、セットする。*/

```
string getIdentifier ( )
    raises (Global::FrameworkErrorSignal);

void setIdentifier (
    in string identifier)
    raises (Global::FrameworkErrorSignal,
        Global::SetValueOutOfRangeSignal,
        Global::DuplicateIdentifierSignal);

/*この MaterialTrackingLocation を持つマシンを返す。*/

Machine getMachine ( )
    raises (Global::FrameworkErrorSignal);

/* MaterialTrackingLocation に保持できるマテリアルユニットを設定する。例:MaterialTrackingLocation がウェーハを保持するためのものなら、ユニットは「ウェーハ」となる。特定のサイズのウェーハであれば、ユニットは「200 mm ウェーハ」となる。*/

void setUnit (
    in string aUnit)
    raises (Global::FrameworkErrorSignal,
        InvalidUnitSignal);

/* MaterialTrackingLocation のユニットを取得する。*/

string getUnit ( )
    raises (Global::FrameworkErrorSignal);

/* MaterialTrackingLocation に Material を追加する。*/

void addMaterialToTrack (
    in AbstractIF::Material aMaterial )
    raises (Global::FrameworkErrorSignal,
        InvalidMaterialSignal,
        MaterialTrackingLocationFullSignal);

/* MaterialTrackingLocation から Material を削除する。*/

void removeMaterial (
    in AbstractIF::Material aMaterial )
    raises (Global::FrameworkErrorSignal,
        InvalidMaterialSignal);

/* MaterialTrackingLocation のキャパシティを予約する。受諾されると、キャパシティのその部分は指定したマテリアルのためにとっておく。キャパシティが予約されると、オペレーションは True を返す。*/

boolean reserveCapacity (
    in AbstractIF::Material capacityToReserve)
    raises (InvalidUnitSignal,
        InvalidMaterialSignal,
        MaterialTrackingLocationFullSignal,
        Global::FrameworkErrorSignal);

/* MaterialTrackingLocation の予約済みキャパシティを解除する。受諾されると、キャパシティの部分は別のマテリアルが使えるよう、開放される。*/

boolean releaseCapacity (
    in AbstractIF::Material capacityToRelease)
    raises (Global::FrameworkErrorSignal);

/* MaterialTrackingLocation の Maximum Capacity(最大キャパシティ)を取得、設定する。*/

void setMaximumCapacity (
    in long maximum)
```

```
        raises (Global::FrameworkErrorSignal,
                Global::SetValueOutOfRangeSignal);

long getMaximumCapacity ( )
    raises (Global::FrameworkErrorSignal);

/* MaterialTrackingLocation の Available Capacity(使用可能キャパシティ)を取得する。Available Capacity とは
Maximum Capacity と、 AllMaterialHeld が返したシーケンスカウントの差である。*/

long getAvailableCapacity ( )
    raises (Global::FrameworkErrorSignal);

/* MaterialTrackingLocation の Reserved Capacity(予約済みキャパシティ)を取得する。Reserved Capacity とは
AvailableCapacity の一部で、まだ到着していないことが分かっているマテリアルのためにとっておく部分である。*/

long getReservedCapacity ( )
    raises (Global::FrameworkErrorSignal);

/*これは MaterialTrackingLocation の使用可能性ステータスに関する問い合わせである。使用可能とは、
MaterialTrackingLocation にマテリアルがもっと入れる使用可能キャパシティがあるということである。*/

boolean hasAvailableCapacity ( )
    raises (Global::FrameworkErrorSignal);
}; // MaterialTrackingLocation
```

6.2.4.4 Port インタフェース(サブセット)

モジュール: MachineAIG

インタフェース: Port

継承インタフェース: Resource

```
interface Port : AbstractIF::Resource {
```

説明: ポートは、マテリアル搬送中に「帰属の変更」が発生する場所を表す。それぞれのポートには、最低一つの関連するマシンがあり、そのポートはそのマシンへの「アクセスポイント」であると考えることができる。

例外: なし。

イベント: なし。

提供サービス:

/*このポートが所属するマシンを返す。*/

```
Machine getMachine ( )  
    raises (Global::FrameworkErrorSignal);  
}; // Port
```

6.2.5 ファクトリオペレーションコンポーネント(サブセット) – 本項には“ファクトリコンポーネント”のインタフェースおよび特定オペレーションで、本コンポーネントスタンダードに参照されているものだけを記載している。それらのインタフェースは、このスタンダードの整合性を図るためここに記載されていて、このコンポーネントの一部として残すことは意図していない。さらに、これは必要なインタフェースを包括的に扱うつもりではなく、マシン抽象インタフェースグループの完全なスタンダードが成立するまで、そのサブセットを参照するため必要となるに過ぎない。マシン抽象インタフェースグループのスタンダードが成立すると、このスタンダードは改訂され、そのインタフェースをここに記載するのではなく直接参照するようになる。

6.2.5.1 MachineRegister インタフェース

モジュール: FactoryOperations

インタフェース: MachineRegister

継承インタフェース: Resource

```
interface MachineRegister : AbstractIF::Resource {
```

説明: MachineRegister には、工場で分かっているマシンのリストを持っており、要求に応じて関連情報を提供する。ファクトリ当たり一つの MachineRegister がある。注:このインタフェースは最終的には取引業者のサービスと入れ替わる可能性がある。

例外:

/*未知のマシンの位置を特定しようとした。*/

```
exception MachineNotFoundSignal {string machineName;;}
```

```
exception MachineRemovalFailedSignal { };
```

公開イベント:

/* マシンがファクトリに追加された、または削除されたことを表すイベント。*/

MachineListChangedEvent

提供サービス:

/* MachineRegister が管理するマシンのセットにマシンを追加する。*/

```
void addMachine (in Machine aMachine)
    raises (Global::FrameworkErrorSignal);

/* MachineRegister が管理するマシンのセットからマシンを削除する。*/

void removeMachine (in Machine aMachine)
    raises (Global::FrameworkErrorSignal,
           MachineRemovalFailedSignal);

/* MachineRegister が管理するマシンすべての順番を返す。*/

MachineSequence allMachines ( )
    raises (Global::FrameworkErrorSignal);

/*指定した名前に対応するマシンを返す。*/

Machine findMachineNamed (in string identifier)
    raises (Global::FrameworkErrorSignal,
           MachineNotFoundSignal) ;

短縮されたサービス: なし。

動的モデル:      なし。

}; // MachineRegister
```

6.3 マテリアル搬送抽象インタフェース — マテリアル搬送抽象インタフェースは、ファクトリレベルのマテリアル搬送を実行するために定義されている。マテリアル搬送マネージャインタフェースのユーザは `TransportJobSupervisor` インタフェースを特化する必要がある。これらのインタフェースは、ファクトリ中ジョブをトラッキングする方法を提供する。

6.3.1 *TransportJobSupervisor* インタフェース

モジュール: `MaterialTransport`
 インタフェース: `TransportJobSupervisor`
 継承インタフェース: `JobSupervisor`

```
interface TransportJobSupervisor : AbstractIF::JobSupervisor {
```

説明: `TransportJobSupervisor` は、特定の `TransportJobSupervisor` の領域の中でマテリアル搬送の要求、トラッキング、コントロールをするため、クライアントが必要とするオペレーションを提供する抽象インタフェースである。マテリアル移動に責任を持つ組織は、`TransportJobSupervisor` を継承し、実装する。

タイプ定義:

```
typedef sequence <Global::Properties> JobSpecSequence;
```

例外:

```
exception UnsupportedDestinationTypeSignal { };
```

公開イベント: なし。

提供サービス:

`TransportJobs` の生成は、継承 `JobSupervisor` インタフェースから `requestJob` オペレーションを使って行う。継承インタフェースに指定した `JobType` プロパティは、すべての `TransportJob` 要求に対する `TransportJobType` に設定しなければならない。搬送ジョブの生成に必要な追加 `Job Specification` プロパティは、表 3 に定義されている。`TransportJob` に指定されたマシンタイプは、AMHS 格納マシン、固定バッファ製造装置または内部バッファ製造装置である。それぞれのマシンタイプは、搬送ジョブのジョブ仕様を定義する際に使用する別々のケースを表しており、これらケースも表 3 に定義されている。