

SEMI E86-0200

CIM フレームワークファクトリ従業員コンポーネントに関する暫定仕様

免責事項: この SEMI スタンドダードは、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。

本仕様は、Global Information & Control Committee で技術的に承認されたもので、North American Information & Control Committee が直接責任を負うものである。現版は 1999 年 10 月 21 日に North American Regional Standards Committee にて承認された。2000 年 1 月にまず www.semi.org で入手可能になり、2000 年 2 月発行に至る。初版は 1999 年 9 月発行。

1 目的

1.1 ファクトリ従業員コンポーネントは製造オペレーションに関わる、資格を持ち、手の空いた人的資源の資格認定および管理をサポートする性能を提供する。

1.2 本文書の技術的内容は、SEMI E81, Provisional Specification for CIM Framework Domain Architecture に記述されている CIM フレームワークドメインの一部であり、資格のあるファクトリ従業員の有用性管理に関連してとりあげるものとする。本仕様は製造実行システムが下記を行うのに必要なインタフェースを提供する:

- 従業員リソースモデルをコンフィグレーションする。
 - － 従業員名 (あるいは別名)、ID、部署、役割など。従業員を必要なファクトリ労働力記録に関連づけるのに識別情報を使ってはいけない場合もある。
 - － 個々の従業員が得意とする技能、およびジョブを実行しデータにアクセスするための認証。
 - － 従業員が最初に割り当てられた他のファクトリリソース (特定のマシンあるいはファクトリエリアなど)。
 - － 各従業員の健康状態。
- 技能メンテナンスタスクおよびそれを実施するためのトリガーを定義する。
- 従業員の技能、資格認証、および割り当てをトラッキングして報告し、それを従業員履歴に記録する。
- 技能メンテナンスのトリガー (技能認定書の期限切れなど) をモニターし、技能メンテナンスジョブの実施を勧告する。
- トリガーによって発生した技能メンテナンスジョブを実行する。
 - － (トレーニングの前後で) 従業員の能力を変える。
 - － 技能メンテナンスジョブ (トレーニング、認定テスト) を実行、モニターし、ジョブの進行状態を報告する。
 - － 技能メンテナンストリガーおよびジョブの結果を従業員の技能メンテナンス履歴に記録する。
- 従業員および技能のパフォーマンスデータ (利用率および有効性) を収集し、報告する。
- 身元の確認および資格認定のため、従業員とシステムセキュリティの関係を管理する。

2 範囲

2.1 具体的には本コンポーネントは下記を行うためのサービスを提供する:

- ファクトリオペレーションに関わる、資格を持ち、手の空いた労働力リソースの割り当てを管理する。
- ファクトリオペレーションのためのトレーニングおよび健康に関する要求条件を管理する。

- 従業員に関する情報(フルネーム, 従業員番号, 部署, シフト, 操作が許されているマシン, 修了した研修コースおよび受信済みの健康診断の記録)の管理をサポートする。
- ファクトリ従業員の技能認定履歴(修了した研修コースおよび受信済みの健康診断)の管理をサポートする。

2.2 ファクトリ従業員コンポーネントは2つのサブコンポーネントに分かれる:

- 従業員管理。製造施設のファクトリオペレーション従業員に関する可用性およびタスクの割り当てをトラッキング, 記録, および管理する。従業員に関する情報, トレーニング, および健康状態の管理もサポートする。
- 技能管理。ファクトリオペレーションに関するトレーニングおよび健康に関する要求条件の管理をサポートする。

2.3 図1には上記の2つのコンポーネントの関係および他の SEMI E81, Provisional Specification for CIM Framework Domain Architecture, 7.2 Functional Partitioning への相互作用が描かれている。

2.4 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。

3 制限

3.1 今後いくつかの分野を完成させる必要があるため, 本仕様は暫定仕様とする。本暫定仕様には不十分な点があり, 完全なスタンダードにするためバロットにかける前にそれに対処しなければならない。不足している点は下記の通りである。

3.2 暫定ステータス

3.2.1 ファクトリ従業員コンポーネントを完全なスタンダードにするためのバロットを実施する前に下記の項目を完成させる必要がある。

- 技能メンテナンスタスク, およびそのトリガーを定義する。
- 技能メンテナンスのトリガー(スキル認定書の期限切れなど)をモニターし, 技能メンテナンスジョブの実施を勧告する。
- 身元の確認および資格認証のため, 従業員とシステムセキュリティの関係を管理する。

3.2.2 今後発行予定のリソース抽象インタフェース定義文書が最初の2つの項目を完成させられると思われる。

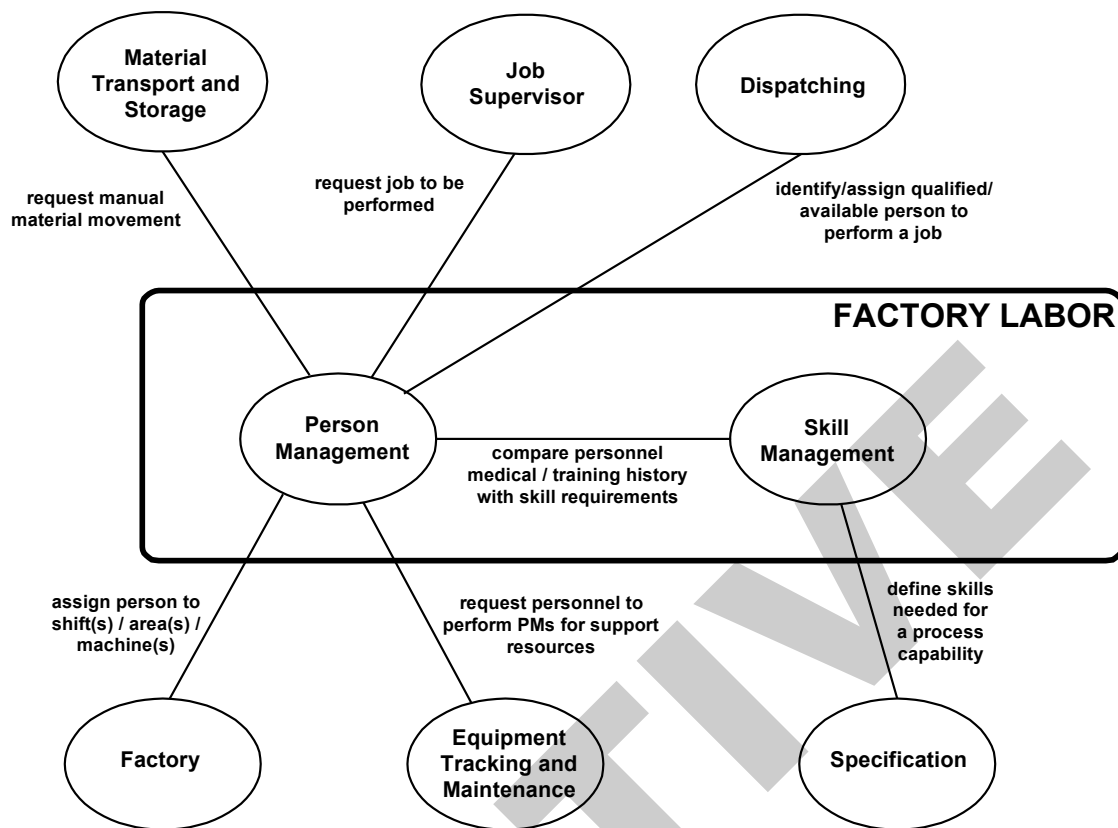


図 1
ファクトリ従業員コンポーネントの関係

4 参照スタンダード

4.1 SEMI スタンダード

SEMI E81 — Provisional Specification for CIM Framework Domain Architecture

4.2 ISO/IEC スタンダード¹

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

4.3 OMG 文書²

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

注: リストに掲載されている文書でもあるいは改訂された文書でもすべて, 引用されたものは適用スタンダードの最新版でなければならない。

¹ ISO Central Secretariat, 1, rue de Varembe, 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

² Object Management Group, Inc., Framingham Corporate Center, 492 Old Connecticut Path, Framingham, MA 01701, 1.508.820.4300, <http://www.omg.org/cgi-bin/doclist.pl>

5 用語

- 5.1 従業員 (*person*) – ファクトリで製造作業に従事する従業員。
- 5.2 従業員の責任 (*person responsibility*) – 「責任」という考え方には、ファクトリの従業員は特定のファクトリオブジェクトを実施する、操作する、あるいはそれにアクセスすることを認められている、という意味がある(が、それを所有しているということは意味しない)。
- 5.3 技能 (*skill*) – 技能とはファクトリ従業員の属性の一つで、その従業員が、プロセス性能を実際に使う際、あるいは他のファクトリオペレーションを実施する際にプロセスリソースあるいはジョブの補助をすることが認められていることを表す。
- 5.4 技能の認定 (*skill qualification*) – 技能の認定とは、ファクトリ従業員があるファクトリオペレーションを実行するのに必要な要求条件の一つ(あるいは全部)を満たしている、ということの意味する。
- 5.5 技能要求条件 (*skill requirement*) – 技能要求条件とは特定の技能を身につけるのに要求されるトレーニングあるいは健康診断である。

6 要求条件

6.1 本項にはファクトリ従業員コンポーネントを構成するインタフェースの仕様が記されている。使用されているモデル表記法は統合モデリング言語 (Unified Modeling Language) (UML) である。UML の説明は UML 表記法ガイドに記されている。各インタフェースの詳細は ISO スタンドラードである IT-ODP-IDL を使って書かれている。ファクトリ従業員コンポーネントの準拠インプリメンテーションは IDL 仕様および UML モデルで使われている意味に関するエレメントを全部サポートしなければならない。IDL 仕様へのマッピングが行われている限り、仕様に従った実装の取り組みは制約されない。

6.2 外部インタフェース, グローバルタイプの定義, グローバルな例外およびイベント

6.2.1 ファクトリ従業員コンポーネント仕様をサポートするのに必要とされるのは、(本コンポーネントの外にある) 下記のインタフェース, グローバルタイプの定義, グローバルな例外, およびイベントである。

6.2.2 抽象インタフェースおよびベースインタフェース

6.2.2.1 ベースインタフェースは CIM フレームワーク仕様で使われる一連のインタフェースを提供する。抽象インタフェースはフレームワークで「継承インタフェース」として使われる抽象インタフェースの階層を提供する。ベースインタフェースおよび抽象インタフェースはインプリメンテーション毎に特殊化されることになっている。

6.2.2.2 本項ではファクトリ従業員コンポーネント仕様で必要とされるベースインタフェースおよび抽象インタフェースを記す。

```
interface NamedEntity {  
};  
  
interface OwnedEntity : NamedEntity {  
};  
  
interface Resource : OwnedEntity : OwnedEntity {  
};  
  
interface ComponentManager : Resource {  
};  
  
interface Job {  
};
```

6.2.3 それ以外の CIM フレームワークインタフェース

6.2.3.1 本項ではファクトリ従業員コンポーネントとの関係を満たすために必要な、本コンポーネント以外の CIM フレームワークコンポーネントにあるインタフェースを記す。

```
interface HistoryCollection { // from Factory Services
};

interface History { // from Factory Services
};

interface MESFactory { // from Factory Management
};
```

6.2.4 グローバルタイプの定義

6.2.4.1 CIM フレームワークでは 2 つの方法で IT-ODP-IDL *typedef* を使っている:

- 複合オブジェクトタイプの別名を定義する(*structs*)。その名前は IDL 仕様で使用できる。
- 意味を追加して、基本オブジェクトタイプの別名を定義する。(例えば、ユニット(*units*)は文字列(*string*)に型定義されていて、その内容は SEMI E5 に書かれている定義に準拠している。)

6.2.4.2 本項ではファクトリ従業員コンポーネント仕様に必要とされるグローバルタイプの定義を記す。

```
typedef sequence<string> stringSequence ;

typedef struct HistoryEvent_body {
    string subject ;
} HistoryEvent ;

struct ulonglong {
    unsigned long low ;
    unsigned long high ;
} ;

typedef ulonglong TimeT ;

typedef TimeT TimeStamp ;

struct IntervalT {
    TimeT lower_bound ;
    TimeT upper_bound ;
} ;

typedef IntervalT TimeWindow ;

typedef struct Shift_Structure {
    string name ;
    // ...
} Shift ;

typedef struct PersonResponsibilitystruct {
    string responsibilityCategory ;
    any responsibility ;
} PersonResponsibility ;

typedef sequence<ProcessCapability> ProcessCapabilitySequence ;
typedef stringSequence CategorySequence ;
typedef sequence<HistoryEvent> HistoryEventSequence ;
typedef sequence<Job> JobSequence ;
typedef sequence<Machine> MachineSequence ;
typedef sequence<Person> PersonSequence ;
typedef sequence<QualificationData> QualificationDataSequence ;
typedef sequence<Resource> ResourceSequence ;
typedef sequence<SkillRequirement> RequirementSequence ;
typedef sequence<PersonResponsibility> ResponsibilitySequence ;
typedef sequence<Skill> SkillSequence ;
```

```
typedef string PropertyName ;
struct Property
{
    PropertyName property_name ;
    any property_value ;
} ;
typedef sequence<Property> Properties ;
```

6.2.5 グローバルな例外の宣言

6.2.5.1 例外とは、オペレーション実行時に異常な状況が発生したことをそのオペレーションのコールをかけたクライアントに通知するのに使われるインフラストラクチャメカニズムである。本項ではファクトリ従業員コンポーネント仕様に必要とされるグローバルな例外の宣言を記す。

```
exception FrameworkErrorSignal {
    unsigned long errorCode ;
    any errorInformation ;
} ;

exception InvalidStateTransitionSignal {
} ;

exception SetValueOutOfRangeSignal {
} ;
```

6.2.6 イベント

6.2.6.1 イベントとは重要な出来事の発生を表す非同期的なメッセージである。例としては状態の変化あるいは新しいオブジェクトの生成などがある。本項ではファクトリ従業員コンポーネント仕様で定義されているイベントを記す。"/"/"あるいは"/"/* */"はコメントを意味する。

```
/* 従業員能力変更イベントの定義 */
const string PersonLifecycleSubject =
    "/ PersonManagement / PersonManager / PersonLifecycle " ;
typedef struct PersonLifecycleFilters_Structure {
    Propertyname ; // "Name", aPerson's name
    PropertyLifecycleEvent ; // "LifecycleEvent", LifecycleState
} PersonLifecycleFilters ;
typedef struct PersonLifecycleEvent_Structure {
    string eventSubject ;
    TimeStamp eventTimeStamp ;
    PersonLifecycleFilters eventFilterData ;
    Properties eventNews ;
    Person aPerson ;
    // Except when "Delete" where value will be nil
} PersonLifecycleEvent ;

/* 従業員能力変更イベントの定義 */
const string PersonCapacityChangedSubject =
    "/ PersonManagement / Person / CapabilityChanged " ;
typedef struct PersonCapacityChangedFilters_Structure {
    Propertyname ; // "Name", aPerson's name
} PersonCapacityChangedFilters ;
typedef struct PersonCapacityChangedEvent_Structure {
    string eventSubject ;
    TimeStamp eventTimeStamp ;
    PersonCapacityChangedFilters eventFilterData ;
    Properties eventNews ;
    Person aPerson ;
} PersonCapacityChangedEvent ;
```

```
/* 従業員状態変更イベントの定義 */
const string PersonStateChangedSubject = "/ PersonManagement / Person / StateChanged" ;
enum PersonState { PersonDefined, PersonOffShift, PersonOnShift, PersonAvailableForWork,
PersonNotAvailableForWork, PersonUnassignedToMachines, PersonAssignedToJobs,
PersonAssignedToMachines, PersonAvailableForMoreAssignment,
PersonAssignmentAtCapacityExceeded, PersonIdleWithJob, PersonBusyWithJob } ;
typedef struct PersonStateChangedFilters_Structure {
    Propertyname ; // "Name", aPerson's name
    PropertyformerState ; // "FormerState", PersonState
    PropertynewState ; // "NewState", PersonState
} PersonStateChangedFilters ;
typedef struct PersonStateChangedEvent_Structure {
    string eventSubject ;
    TimeStamp eventTimeStamp ;
    PersonStateChangedFilters eventFilterData ;
    Properties eventNews ;
    Person aPerson ;
} PersonStateChangedEvent ;

/* 従業員シフト変更イベントの定義 */
const string PersonShiftChangedSubject = "/ PersonManagement / Person / ShiftChanged" ;
typedef struct PersonShiftChangedFilters_Structure {
    Propertyname ; // "Name", aPerson's name
    PropertyfromShift ; // "FromShift", former shift
    PropertytoShift ; // "ToShift", new shift
} PersonShiftChangedFilters ;
typedef struct PersonShiftChangedEvent_Structure {
    string eventSubject ;
    TimeStamp eventTimeStamp ;
    PersonShiftChangedFilters eventFilterData ;
    Properties eventNews ;
    Person aPerson ;
} PersonShiftChangedEvent ;
```

6.3 従業員管理コンポーネント

6.3.1 このコンポーネントは CIM システムに対するファクトリ従業員(従業員インタフェース)を定義する。従業員は製造環境にあるリソースと見なされ、ファクトリエリアに割り当てられ、詳細なトレーニング記録および健康診断記録を持っている(トレーニング履歴および健康診断記録によって従業員はある技能を実行する資格が与えられる)。例えば、オペレータは定期的に特定のトレーニングあるいは医療的なテストを受けなければならないという要求条件があるかもしれない。資格の認定には要求されたトレーニングが指定された期限までに行われたという確認も行われるだろう。関連技能は従業員が資格認定を受けることによって発揮できるようになったプロセス能力を特定する。いかなる場合であれ、他のファクトリリソースあるいはオペレーションと関連した一つ以上のファクトリジョブに従業員を割り当てることができる。

6.3.2 従業員管理のためのコンポーネント情報モデルは図 2 に描かれている。

Error! Objects cannot be created from editing field codes.

図 2

従業員管理コンポーネントの情報モデル

6.3.3 *PersonManager* インタフェース

インタフェース: **PersonManager**

継承インタフェース: **ComponentManager**

説明: これは従業員管理コンポーネントと CIM 環境の残りの部分の間にあるインタフェースである。**PersonManager** はファクトリ内の従業員のインスタンスを生成し、その従業員を管理する(シフト、ジョブ割り当ておよびステータス情報)。ファクトリの **PersonManager** の数に限度はない。

例外:

/* このシグナルはすでに存在している従業員識別子を使って従業員を生成しようとした際に発生する。*/

```
exception PersonDuplicateSignal { string identificationNumber ; } ;
```

/* このシグナルは引数で指定された従業員の位置を突き止められないために検索オペレーションが失敗した際に発生する。*/

```
exception PersonNotFoundSignal { string identificationNumber ; } ;
```

/* このシグナルは引数で指定された従業員が **PersonManager** に割り当てられていないために削除オペレーションが失敗した際に発生する。*/

```
exception PersonNotAssigned { Person unknownPerson ; } ;
```

```
exception PersonRemovalFailedSignal { } ;
```

公開イベント: **PersonLifecycleEvent**

提供されるサービス:

/* ユニークな識別子を持つ従業員のインスタンスを生成し、それを **PersonManager** が管理するリストに追加する。*/

```
Person createPersonWithIdentifier ( in string identificationNumber)
    raises (FrameworkErrorSignal, PersonDuplicateSignal) ;
```

/* 引数で特定された従業員を **PersonManager** が管理するリストから削除する。*/

```
void removePerson ( in Person aPerson)
    raises (FrameworkErrorSignal, PersonRemovalFailedSignal,
        PersonNotAssignedSignal) ;
```

/* **PersonManager** が管理する従業員を返す。*/

```
PersonSequence allPersons ( ) raises (FrameworkErrorSignal) ;
```

/* ユニークな識別子を使って従業員のインスタンスを検索する。従業員が見つからない場合は例外を発生させる。*/

```
Person findByIdentifier ( in string identificationNumber)
    raises (FrameworkErrorSignal, PersonNotFoundSignal) ;
```

/* 指定されたシフトにログオンされている従業員をすべて返す。*/

```
PersonSequence allOnShiftPersons ( in string aShift ) raises (FrameworkErrorSignal) ;
```

/* 仕事に従事できる従業員をすべて返す。*/

```
PersonSequence allAvailableForWorkPersons ( ) raises (FrameworkErrorSignal) ;
```

/* 仕事に従事できない従業員をすべて返す。*/

```
PersonSequence allNotAvailableForWorkPersons ( ) raises (FrameworkErrorSignal) ;

/***** 従業員／マシンの関係を管理する。*****/

/* あるプロセスあるいはマテリアル搬送装置を操作する資格を持っている従業員を返す。従業員がマシンを操作する資格を持っているとされるのは、そのマシンが従業員のスキルリストに載っている場合、あるいは従業員の研修コース履歴および健康診断記録が、当該マシンに関する技能管理コンポーネントで定義されている要求条件を満たす場合である。このメソッドでは従業員の手が空いていてタスクを実際に実行できるかどうかはチェックしない。*/

PersonSequence allPersonsWithSkill ( in Skill aSkill ) raises (FrameworkErrorSignal) ;

/* 手が空いていて指定されたマシンを操作できる(その時点のシフトで「勤務中」である)従業員を返す。このサービスでは従業員はマシンを操作する資格を持っているものとみなす。*/

PersonSequence allPersonsAvailableForMachine ( in Machine aMachine )
    raises (FrameworkErrorSignal) ;

/* その時点で指定されたマシンに割り当てられている従業員をすべて返す。*/

PersonSequence allPersonsAssignedToMachine ( in Machine aMachine )
    raises (FrameworkErrorSignal) ;

/* その時点でマシンに割り当てられている従業員をすべて返す。*/

PersonSequence allAssignedToMachinesPersons ( ) raises (FrameworkErrorSignal) ;

/***** 従業員／マシンの関係を管理する。*****/

/* あるプロセスあるいはマテリアル搬送ジョブを実行する資格を持つ従業員を返す。従業員がジョブを実行する資格を持っているとされるのは、従業員のスキルリストでそのジョブに関する資格が認定されている場合、あるいは従業員の研修コース履歴および健康診断記録が当該ジョブに関する技能管理コンポーネントで定義されている要求条件を満たす場合である。このメソッドでは従業員の手が空いていて実際にタスクを実行できるかどうかはチェックしない。*/

PersonSequence allPersonsQualifiedForJob ( in Job aJob )
    raises (FrameworkErrorSignal) ;

/* 手が空いていて(その時点のシフトで「勤務中」で)プロセスあるいはマテリアル搬送ジョブを実行できる従業員を返す。このサービスでは従業員はジョブを実行する資格を持っているものとする。*/

PersonSequence allPersonsAvailableForJob ( in Job aJob )
    raises (FrameworkErrorSignal) ;

/* その時点で指定されたジョブに割り当てられている従業員を返す。*/

PersonSequence allPersonsAssignedToJob ( in Job aJob ) raises (FrameworkErrorSignal) ;

/* その時点でジョブに割り当てられている従業員を返す。*/

PersonSequence allAssignedToJobsPersons ( ) raises (FrameworkErrorSignal) ;

契約サービス:                なし
動的モデル:                  継承インタフェースと同じ(コンポーネントマネージャ)
```

6.3.4 Person インタフェース

インタフェース: Person

継承インタフェース: Resource