



## **Recipe Management System: Common Requirements and Benchmark 2007**

**International SEMATECH Manufacturing Initiative  
Technology Transfer #08054930A-TR**

**Advanced Materials Research Center, AMRC, International SEMATECH Manufacturing Initiative, and ISMI** are servicemarks of SEMATECH, Inc. **SEMATECH** and the **SEMATECH** logo are registered servicemarks of SEMATECH, Inc. All other servicemarks and trademarks are the property of their respective owners.

**Recipe Management System: Common Requirements and Benchmark  
2007  
Technology Transfer #08054930A-TR  
International SEMATECH Manufacturing Initiative  
May 9, 2008**

**Abstract:** This specification provides a benchmark of current practices and a roadmap for the future of factory recipe management systems (RMS). The document examines recipe management and its effect on the factory system. It defines the scope and functional requirements of the RMS and examines how recipe management is reflected in other parts of the factory system. It establishes a set of common requirements with which International SEMATECH Manufacturing Initiative (ISMI) members can drive their own systems forward and influence commercial RMS suppliers.

**Keywords:** Advanced Process Control, Automation, Business Trends, Data Management Standards, Manufacturing Management, Recipe Management

**Authors:** Lance Rist

**Approvals:** Steve Fulton, Project Manager  
Olaf Rothe, Program Manager  
Scott Kramer, Director, ISMI  
Laurie Modrey, Technology Transfer Team Leader



## Table of Contents

1	EXECUTIVE SUMMARY .....	1
	1.1 Purpose.....	1
	1.2 Scope.....	2
2	RELATED RESOURCES .....	2
	2.1 ISMI Resources.....	2
	2.2 SEMI Standards .....	2
	2.3 Other Documents .....	3
3	DEFINITION OF TERMS AND ACRONYMS .....	3
4	APPROACH.....	4
5	RMS OVERVIEW.....	5
6	RMS REQUIREMENTS.....	7
	6.1 General RMS Requirements .....	8
	6.2 Recipe Catalog .....	8
	6.3 Factory Recipe Store.....	11
	6.4 Equipment Recipe Store Management.....	12
	6.5 Recipe Integrity.....	14
	6.6 Version Management .....	15
	6.7 Change Control .....	16
	6.8 Authentication and Authorization.....	16
	6.9 Parameter Management.....	17
	6.10 Shared Recipe Components .....	17
	6.11 Report Generation .....	18
	6.12 Recipe Component Viewing and Modification.....	18
	6.13 Requirements Table .....	19
7	RECIPE MANAGEMENT EFFECTS ON THE FICS.....	21
	7.1 Scheduling.....	22
	7.2 Dispatching .....	22
	7.3 Process Specification Management.....	22
	7.4 Durable Tracking .....	23
	7.5 Equipment Maintenance Tracking .....	23
	7.6 Data Collection Plan Management.....	23
	7.7 Data Collection .....	23
	7.8 SECS/GEM Host.....	23
	7.9 Process Control .....	24
8	FUTURE VISION FOR FACTORY RECIPE MANAGEMENT.....	24
	8.1 Trends That Will Affect Recipe Management.....	24
	8.1.1 Feature Shrink .....	24
	8.1.2 Next Generation Processing.....	25
	8.2 RMS in Transition.....	26
9	CONCLUSION .....	27
	APPENDIX A – MEMBER SURVEY RESPONSES SUMMARY.....	28
	APPENDIX B – SUPPLIER INVESTIGATION .....	31
	B.1 Recipe Management Systems Suppliers .....	31

**List of Figures**

Figure 1 Typical Current Environment for Recipe Management .....5  
Figure 2 Anticipated Future Environment for Recipe Management .....7  
Figure 3 Factory Components .....21

**List of Tables**

Table 1 Requirements Table .....19  
Table A-1 Member Survey Responses Summary.....28  
Table B-1 Survey Responses.....32

## **1 EXECUTIVE SUMMARY**

Equipment recipes represent significant intellectual property to device makers. They are the means of defining how processing will be done in each of the hundreds of steps required to manufacture finished wafers. Management of these recipes is a growing challenge in today's semiconductor factories. As processes evolve and change more rapidly, the number of equipment recipes has increased dramatically. Process control has, in some cases, mandated a new, often lightly controlled variation of the recipe each time it is used. More sophisticated processing equipment requires more sophisticated recipes; e.g., multi-part recipes have replaced single files, recipe parameters must be settable at run time to supplement the base recipe definition, and recipe size has increased to accommodate added logic, data, images, and more.

Existing factory recipe management systems (RMS) struggle to cope with these increasing challenges. In response, many device makers are modifying (or planning to modify) their approach to recipe management by redesigning their systems to be more sophisticated and to provide more functionality.

Changes to equipment support for recipe management have already been realized in the SEMI standards and will begin to appear on the shop floor in the near future (SEMI E139, Recipe and Parameter Management (RaP)). This will have a dual effect on recipe management. Where RaP is available, it will be easier to accomplish tasks heretofore difficult or impossible. However, during the transition, device makers may be faced with environments that include both the new and the old approach to recipe management on their equipment.

Requirements for factory-level RMS have traditionally been individually determined by each device maker and/or RMS supplier. The results have varied widely from simplistic, per-tool extensions of the SECS/GEM Host software to more sophisticated centralized systems. Goals vary from simply making sure the equipment recipe is present on the tool to providing off-tool storage and configuration management to advanced process control support. More recently, these varying goals have been converging to where most apply to nearly all device makers.

The logical conclusion is that the definition of a common set of requirements for RMS is beneficial and cost-effective for International SEMATECH Manufacturing Initiative (ISMI) members and for the semiconductor industry in general. Such a definition can help device makers consider areas of function they might otherwise ignore. It can also lead to more uniform requirements for commercial RMS suppliers across their customer base. Uniform requirements tend to lead to higher quality and lower cost as the supplier's scope can be narrowed and the development cost can be shared among more customers.

### **1.1 Purpose**

The purpose of this document is to clarify the scope and content of the factory recipe management system. It provides a benchmark of current practices and a roadmap for the future.

This document establishes a common set of requirements for factory recipe management systems that is intended to

- Provide ISMI members guidance in designing or specifying factory recipe management systems
- Provide suppliers of RMS applications more uniformity in the requirements of their various customers

- Establish a common direction for future recipe management systems that meet the needs of next generation semiconductor factories

ISMI members are expected to use this common requirements set to drive their own recipe management systems forward to meet the needs identified and anticipated here. This document also serves as the focal point for further discussions and clarifications within the industry on the topic of RMS.

## 1.2 Scope

This document examines recipe management and its effect on the factory system. It defines the scope and functional requirements of the RMS. It also examines how recipe management is reflected in other parts of the factory system.

This document is not intended to specify RMS design nor a standard interface from the RMS to other factory applications.

Both current and anticipated future requirements for recipe management are included, as well as the transition between them. Future requirements were targeted at the 3–5 year time period, including the anticipated widespread use of RaP, but they also include support for the ITRS out to its current time horizon.

It is understood that the user's factory system approach will determine whether certain requirements must be met within the RMS or in other factory applications. This freedom is incorporated, while the specification urges movement toward a more common scope in future designs.

## 2 RELATED RESOURCES

### 2.1 ISMI Resources

*Recipe and Parameter (RaP) Management Supplier Guidance*, ISMI, Technology Transfer 06104801A-ENG.

*Recipe and Parameter Management (RaP) Usage Scenarios*, ISMI, Technology Transfer #08054929A-TR.

*RaP Reference Implementation (RRI)*, [software application](#).<sup>1</sup>

*“ISMI Next Generation Factory Vision,” ISMI, Unified 450 mm/300 mm Prime Factory Guidelines Workshop*, July 2007.<sup>2</sup>

*ISMI Industry Briefing – Next Generation Factory*, ISMI, December 2007.<sup>3</sup>

### 2.2 SEMI Standards

SEMI standards are available for purchase at <http://www.semi.org/standards>.

SEMI E30 – *Generic Equipment Model (GEM)*.

---

<sup>1</sup> <http://ismi.semiatech.org/rri>

<sup>2</sup> <http://ismi.semiatech.org/wafersize/index.htm>

<sup>3</sup> <http://ismi.semiatech.org/wafersize/index.htm>

SEMI E40 – *Standard for Processing Management*.

SEMI E139 – *Specification for Recipe And Parameter Management (RaP)*.

### 2.3 Other Documents

[International Technology Roadmap for Semiconductors](#), 2007 Edition, ITRS, 2007.<sup>4</sup>

## 3 DEFINITION OF TERMS AND ACRONYMS

**Equipment Recipe** – A specification used by a production equipment to direct its execution of a task. See also, “Recipe.” An equipment recipe may consist of one or more parts (recipe components). When a recipe has multiple parts, the *collection* of recipe components needed for a specific task is considered the recipe.

**Equipment Recipe Store** – The collection of recipe components stored on a piece of production equipment.

**Factory Information and Control System (FICS)** – Software system that controls manufacturing in a factory. The FICS includes the Manufacturing Execution System (MES), automation software that directly controls factory equipment and material handling, as well as off-tool process control systems. Recipe management systems are a part of the FICS.

**Golden Copy** – The official master version of a specification (e.g., equipment recipe), known to be in its intended or correct form.

*International Technology Roadmap for Semiconductors (ITRS)*<sup>4</sup>.

**Master PDE** – The PDE at the top of the recipe hierarchical structure. In a multi-part recipe, this is often the “sequence” recipe that describes how the substrate will progress through the equipment processing chambers (see SEMI E139).

**Process Definition Element (PDE)** – An executable specification for all or part of an activity or process on a piece of equipment (see SEMI E139). A PDE is a recipe component.

**Recipe and Parameter Management (RaP)** – Equipment automation capability to support management of equipment recipes (process programs). See SEMI E139.

**Recipe** – A pre-planned and reusable set of instructions and settings that specify how a task is to be performed. A recipe may consist of multiple recipe components. In the context of this document, recipes are specifications for activities related to production equipment. See also “Equipment Recipe” and “RMS Recipe.”

**Recipe Component** – A separate, identifiable part of a recipe. When a recipe consists of multiple parts, each of the parts is referred to as a recipe component. Recipe components have individual identifiers and are managed as separate entities. See also PDE.

**Recipe Parameter** – A recipe setting that can be modified independent of the recipe itself in messaging directly to the equipment (e.g., during Process Job creation, see SEMI E40). Recipe parameters allow for process variation (usually within limits) without changing the recipe itself.

---

<sup>4</sup> <http://www.itrs.net/Links/2007ITRS/Home2007.htm>

**Recipe Management System (RMS)** – An application for managing equipment recipes within a factory. This factory-level system may take a distributed or a monolithic approach to providing a homogeneous system for factory recipe management.

**RMS Recipe** – Complete specification for a production activity within the factory system. The RMS Recipe includes the recipe components of the equipment recipe, additional recipe components to be consumed by other factory applications (e.g., SECS/GEM Host), and related information (descriptive and status) maintained by the RMS.

**Semiconductor Equipment Communication Standard (SECS)** – Communication protocol for semiconductor factory automation. See SEMI E05 for more detail.

**SECS/GEM Host** – The software application that communicates directly with production equipment and facilitates factory operations involving that equipment. This is sometimes alternatively referred to as the equipment interface (EI), station controller, cell controller, machine manager, etc. This definition does not seek to define these as being similar in scope, but rather to provide a term that can be used to commonly refer to this general concept.

## 4 APPROACH

These common requirements were created mid-year 2007, based on inputs from industry experts in recipe management and on forward-looking bodies including the ITRS and the groups evaluating the transition to 450 mm manufacturing.

Recipe management experts were drawn from two sources:

- ISMI member companies that provided access to their internal recipe management experts
- Commercial RMS suppliers that were determined to have stand alone and full-featured products

A project working group consisting of ISMI personnel and member company volunteers collaborated to create an RMS survey. This survey was used to gather information from both ISMI member company experts (see Appendix A) and commercial RMS suppliers (see Appendix B). The survey was first distributed to these groups for written input. To ensure a complete understanding of the results, a telephone interview was conducted with each of the respondents.

Additional information was gathered through study of the following:

- The most current work of the ITRS (including discussions with some participants)
- Public presentation materials from the ISMI 450 mm program
- Public presentation materials from JEITA on next generation technology<sup>5</sup>
- The SEMI E139 RaP standards and related presentation materials

---

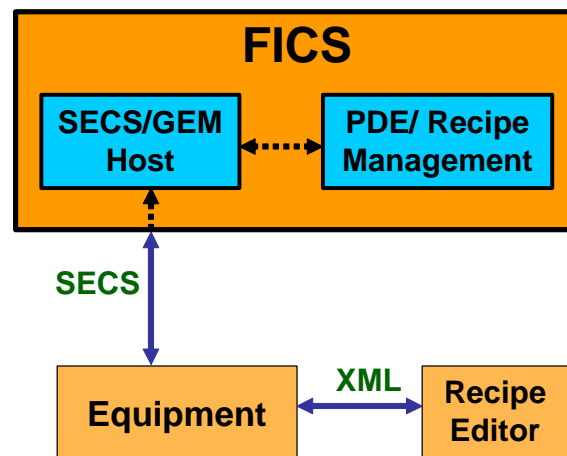
<sup>5</sup> Information based on handouts that were available at public workshops in 2007. Material is not known to be available on the Internet.

This document was created by ISMI using its best engineering judgment and further modified and improved through review, feedback, and discussion among the project working group members.

## 5 RMS OVERVIEW

Recipe management at the factory level is responsible for most aspects related to the specification of how the production equipment should perform their operations. The scope of the recipe management system has grown over time. Initially, recipe management was just a function within the SECS/GEM Host to backup the recipes from the equipment into a host-side file directory. As the manufacturing process became more complex, the RMS became a separate entity with far more sophisticated capabilities. Ownership of the recipes has shifted from the equipment-SECS/GEM Host level to the factory level. This shift has led to more uniform management of recipes within a given factory. However, such organically driven development resulted in variations as individual factories emphasized different functional areas relating to their own most urgent needs. The result is the current heterogeneous set of recipe management systems.

Figure 1 shows a typical current environment for recipe management with all host communications to the equipment flowing through the command and control connection using SECS protocol. Some off-tool recipe editors exist, using a multi-client protocol such as XML/SOAP over HTTP.



**Figure 1 Typical Current Environment for Recipe Management**

As a result of the research for this document, the following list of capability areas for recipe management was created. This list helps to understand the potential scope of an RMS:

- Recipe Catalog – Tracking of information and status related to recipes and recipe components within the factory.
- Factory Recipe Store – Storage and retrieval of recipe components for use by the factory.

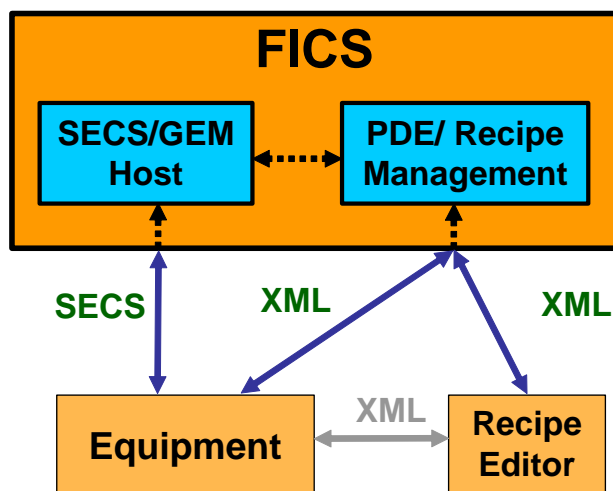
- Equipment Recipe Store Management – Tracking and management of information about the recipes stored on the production equipment in the factory.
- Recipe Integrity – Activities intended to 1) ensure that each recipe component on the equipment and used during processing is a true replica of the “golden copy” designated by the factory system and to 2) ensure that the combination of recipe components used by the equipment is the exact set intended by the factory to make up the recipe.
- Version Management – Management of changes to recipes and recipe components that facilitates the proper version of a recipe being used at the proper time.
- Change Control – Enforcement of rules relating to approval of changes in the RMS and the recipes and data within by designated authorized users.
- Authentication and Authorization – The process of allowing only certain users and applications to access the RMS and limiting each to only the appropriate access and functions.
- Parameter Management – Management of the description of what parameters can/must be provided to modify the settings of a recipe for a specific processing run and any related rules (e.g., parameter limits and the allowed sources for parameter values).
- Shared Recipe Components – A provision for the reuse of recipes and recipe components among different processes; different equipment; and, for recipe components, among different recipes.
- Recipe Component Viewing and Modification – The ability to make changes to recipes and recipe components. These changes may be as simple as automatically renaming a recipe component to conform to factory naming conventions or as complex and changing settings with the executable part of the recipe.

These recipe management capability areas and a few others (such as “general” and “reporting”) are used to organize the requirements described Section 6. The category groupings were created to enhance understanding by describing the functional needs; they are not intended to imply an architecture or implementation.

An additional consideration for the RMS is its ability to directly cause action within the factory. Early recipe management systems were completely passive, responding to requests (“store recipe” or “change recipe state”), but relying completely on other systems to initiate any action. In some current systems and most future systems, the RMS will take a more active role in the factory. For example, it may react to a scheduler’s prediction of the processing of a certain lot on a specific equipment by initiating the download of the needed recipe to that equipment. It will also be more active in monitoring and controlling which recipes are stored on a piece of equipment. The requirements in this document reflect this anticipated evolution of RMS.

Figure 2 shows the anticipated future environment for recipe management. The command and control interface using SECS protocol will continue to check for recipe presence and integrity at job creation time. Management of the recipe store will move to the recipe management system. The RMS will have direct communication to the equipment for recipe download. Off-tool recipe editors will become more prevalent and migrate from direct tool communication to be linked to the RMS.

Several recipe management-related activities occur in factory system modules other than the RMS. For example, the SECS/GEM host must interact with the RMS to ensure that the correct recipe is downloaded and the correct parameter values are supplied. Less obvious is the need for the Process Specification manager to understand which recipes must be used in each processing context and be up-to-date as new versions of those recipes are created. These are discussed in more detail in Section 7.



**Figure 2 Anticipated Future Environment for Recipe Management**

## 6 RMS REQUIREMENTS

The functions defined below are common requirements for a recipe management system as determined by ISMI member company consensus. While this document should describe the major portion of RMS requirements for all, it is expected that individual device makers will have some additional requirements that add features and detail to the broader requirements in this document. The requirements are summarized in Table 1.

Each requirement is classified as “Core” or “Active.” A core requirement is central to an RMS; it is normally passive (see Active vs. Passive discussion in Section 5). An RMS is expected to support all of the core requirements. The second classification is “Present” or “Future.” For further explanation of both classifications, see Section 6.13.

Certain terminology must be clear to fully understand these requirements. In this document, an *equipment recipe* is a set of one or more recipe components that are consumed directly by the equipment. These components of the equipment recipe are individually downloaded to the equipment. In traditional SECS-II terminology, each of these downloadable files is referred to as a recipe. This was reasonable when only single part recipes were used. Today, multi-part recipes are very common; consequently, the SECS-II definition of “recipe” is not used in this document.

For the RMS, the recipe is more than just the equipment recipe. It also includes data attributes and status values important to describing and tracking a recipe. In addition, there may be other recipe components that are consumed by factory applications rather than by the equipment. For example, a recipe component might contain equipment setup instructions used by the SECS/GEM Host to set equipment constant values, check equipment status, etc. To emphasize this enlarged scope, the recipe managed by the RMS is called an *RMS recipe*.

## 6.1 General RMS Requirements

The RMS shall provide a user interface that allows an authorized user to perform all actions available for the system and to set all data as appropriate.

- **<GR01 – User Interface Access>**

All actions available as services via a communications interface must also be provided through the user interface.

## 6.2 Recipe Catalog

The RMS needs to create, manage, archive, and track RMS recipes, recipe components, and associated data used in recipes. This system needs to manage the access of recipes, control any changes, and record these changes through their lifecycle.

- **<RC01 – Recipe Definition>**

The definition of a recipe in the RMS shall include all managed aspects of the RMS recipe including the recipe components, recipe status, and additional attribute values as needed by the system and the user.

- These recipe aspects are discussed in other areas of this document.

- **<RC02 – Recipe Creation>**

The RMS shall provide a means for creating an RMS recipe, including all required parts of the recipe definition.

- **<RC03 – Get Recipe Information>**

The RMS shall provide a service<sup>6</sup> to return to an authorized requestor selected information about an RMS recipe, including a list of recipe components, other recipe descriptive data, and recipe status.

- **<RC04 – Recipe Access>**

The RMS shall provide a service to return to an authorized requestor the set of recipe components included in a specified RMS recipe from the factory recipe store.

- If multiple RMS recipes are returned, there needs to be a way to separate the parts of one recipe from those of another. It is recommended that a manifest list of which components belong to each recipe be included.

- **<RC05 – Recipe relationships>**

The RMS shall have knowledge of recipe component relationships (multi-part recipes) and shall manage them in a defined manner.

- See also Version Management (Section 6.6) and Change Control (Section 6.7).

- **<RC06 – Recipe Descriptive Data>**

The RMS shall store descriptive data about each RMS recipe:

- Recipe description (text)

---

<sup>6</sup> The requirement of a “service” should not be interpreted as a specific service limited to this need. A single service may be implemented to meet multiple service requirements.

- Recipe type
- The list of components and related files that make up this recipe
- Parameters (see Section 6.9)
- Product/product family
- Version
- Tools on which the recipe will be used a) for which is it approved to use, b) on which can it run
- Process context data: Process context is used to associate an instance of processing with the equipment recipe needed for that situation. Context varies slightly from one factory to another, but tends to identify the equipment, the product, and the step in the process flow.
- User-defined information: The user should be able to customize the RMS to add additional information.

- **<RC07 – Recipe Component Descriptive Data>**

The RMS shall store and provide access to recipe component descriptive data and other associated information:

- Recipe component description
- Recipe component type
- Version
- Parameters
- Tools on which the recipe component can be used a) for which is it approved to use, b) on which can it run
- User-defined data: The user should be able to customize the RMS to add recipe component attributes.
  - An example of user-defined data is an “IP protection flag.” This flag might be used to indicate that a recipe component contains sensitive information that must be protected. When the SECS/GEM Host sees an IP protection flag set, it deletes the recipe from the equipment immediately after it is used.

- **<RC08 – Recipe Lifecycle Status>**

The RMS shall provide and manage a recipe lifecycle status for each RMS recipe and for each recipe component known to the system.

- The status of the RMS recipe or recipe component shall be described as a state model with defined states and specifically allowed transitions between those states.
- The user shall be allowed to customize the states and transitions (including allowed uses of recipes or recipe components in a given state).
- This document does not specify the states to be included. However, those states shall include concepts comparable to the following:
  - Definition: The RMS recipe has been created, but the definition is not yet complete.

- Under Development: The components of the RMS recipe are being created, modified, tested, etc., to prepare them for qualification.
- Qualification: The RMS recipe is being qualified for production use.
- In Approval Cycle: The RMS recipe is in the approval/signoff process.
- Active: The RMS recipe is ready and available for use (e.g., production, maintenance, or equipment qualification).
- Inactive: The recipe was previously Active, but is no longer available for use.
- Archived: The recipe has been archived and is no longer available.

- **<RC09 – Archive Recipe>**

The RMS shall provide the ability to archive and subsequently restore an RMS recipe, including its related recipe components:

- Archiving an RMS recipe means extracting the recipe components and any associated information needed to recreate the RMS recipe definition in a form that can be stored off-line/off-site. The recipe name and appropriate descriptive data are retained in the catalog for informational purposes. The related recipe components, if not used by other (non-archived) RMS recipes and not already archived, are removed from the Factory Recipe Store.
- Individual recipe components can also be archived (see [FRS05](#)).
- Restoration may depend on the availability of the archive volume. Archived volumes may be stored off-site and require physical retrieval.
- The state of the restored RMS recipe will depend on the user's defined logic (see [RC08](#)).

- **<RC10 – Recipe Actions>**

The RMS shall use the recipe status to manage what actions can be applied in the various states and the allowed transitions for any given state (see also RC08 as well as Authentication and Authorization, Section 6.8).

- **<RC11 – Get Recipe List>**

The RMS shall provide a service to return to authorized requestors a list of the RMS recipes defined in the Recipe Catalog. The returned list shall be filtered automatically to include only those applicable to the requestor. The returned list shall be additionally filtered based on requestor selections.

- Requestor filtering should be tied to known attributes for the recipe component (e.g., status, target equipment, target product, creation date, author, etc.).
- This service can be used to identify a recipe by process context. The requestor can set the filter for the desired context attributes (see [RC06](#)).

- **<RC12 – Recipe Browser>**

The RMS shall provide an authorized user the capability to view RMS recipe content.

- It should support displaying all aspects of an RMS recipe:
  - Descriptive data

- List of recipe components
- Recipe lifecycle status
- Version history
- Change history

### 6.3 Factory Recipe Store

The factory must have a secure place at the factory level to store copies of all recipe components to ensure that these are preserved and available when needed. Storage at the factory level gives the device maker direct control of this capability. This “central” storage may be distributed, if desired, but should be centrally managed.

- **<FRS01 – Store Golden Copy>**

The RMS shall store designated recipe components in a secure manner that ensures that the content is preserved and maintained.

- **<FRS02 – Recipe Component Access>**

The RMS shall provide a specified component or set of components upon request by an authorized requestor.

- **<FRS03 – Recipe Component Upload>**

The RMS shall provide a service to authorized requestors to introduce into the factory recipe store a recipe component not currently present in the repository.

- For a multi-part recipe, the full “recipe” is not typically introduced as a unit, since the RMS definition of a recipe includes more than the equipment recipe. Each recipe component is added to the Factory Recipe Store. Recipes are created within the RMS independent of recipe component upload.

- **<FRS04 – Get Recipe Component List>**

The RMS shall provide a service to return to authorized requestors a list of the RMS recipe components available in the Factory Recipe Store. The returned list shall be filtered automatically to include only those applicable to the requestor. The returned list shall be filtered based on requestor selections.

- Requestor filtering should be tied to known attributes for the recipe component (e.g., status, target equipment, creation date, author, recipe type, etc.).
- For a RaP PDE, the requestor filtering should include PDEheader attributes.

- **<FRS05 – Archive Recipe Component>**

The RMS shall provide the ability to archive and subsequently restore selected recipe components.

- Archiving means extracting a recipe component and related information in a form that can be stored off-line/off-site and removing the recipe component from the Factory Recipe Store. The recipe component name and descriptive data are retained in the catalog for informational purposes.
- Note that archiving a recipe component may also occur as part of archiving a recipe (see [RC09](#)). These two requirements refer to the same archive.

- **<FRS06 – Recipe Component>**

The RMS shall store as a recipe component any data set designated by the user to be used to execute a recipe.

- **<FRS07 – Delete Component>**

The RMS shall provide a protected method to delete a recipe component.

- Deleted components shall be retained until all links and references are cleared. For example, a component required by an existing RMS recipe is not removed.
- Recipe components that are no longer needed will typically be archived rather than deleted. The delete method would be expected to require a special level of authorization.

- **<FRS08 – Never Replace Recipe Component>**

The RMS shall not allow a recipe component to be replaced by another recipe component.

- An exception to this requirement is restoring a corrupted recipe component from a known good copy.
- The new recipe component should instead be considered a new version of the existing one (see Version Management).

- **<FRS09 – Export/Import Recipe Components>**

The RMS shall be capable of importing and exporting recipe components to facilitate transfer to/from a different RMS (e.g., to a different factory).

## 6.4 Equipment Recipe Store Management

At the factory level, there needs to be a method to track and manage recipe components stored at the equipment. This tracking is independent of whether a copy of this component is kept in the factory recipe store. However, these recipe components must be known to the RMS (see Section 6.2).

- **<ERS01 – Equipment Recipe Store List>**

The RMS shall be able to store a list of which recipe components are stored on each equipment.

- Tracking the equipment recipe store is effective only if the equipment can ensure recipe integrity (e.g., that no change to a recipe can occur).

- **<ERS02 – Equipment Recipe Change Event>**

The RMS must update the equipment recipe store based on events sent from the equipment or from an intermediary. The RMS shall provide a service to facilitate the update of the equipment recipe store list by an authorized application based on such equipment events.

- Events (event data) tell the RMS when a recipe component has been added or removed from an equipment's recipe store. In most cases, the event will be delivered from the equipment to a SECS/GEM Host, which will forward the information to the RMS.

- **<ERS03 – Equipment Recipe Store List Update>**

The RMS shall provide a service to allow an authorized application to update the Equipment Recipe Store list for an equipment.

- The entire recipe list from the equipment is expected to be supplied. In this way, the update will imply both additions and deletions of recipe components on the equipment.

- **<ERS04 – Equipment Recipe List Request>**

The RMS shall be able to request a full recipe directory list from an equipment, whenever needed.

- Note that this request may need to be sent to an intermediary (such as the SECS/GEM Host), rather than directly to the equipment.

- **<ERS05 – Equipment Recipe Component Delete Flag>**

The RMS shall maintain a flag to indicate that a recipe component needs to be deleted from the equipment.

- Note that this flag will be set or cleared based on change notifications according to [ERS02](#), [ERS03](#), and [ERS04](#).

- **<ERS06 – Equipment Recipe Component Download Flag>**

The RMS shall maintain a flag to indicate that an equipment recipe component needs to be downloaded.

- The implementer of the RMS might choose to combine the download flag and delete flag ([ERS05](#)) into a single status that indicates that the recipe should or should not be present on the equipment. These two requirements are not intended to dictate the design.
- Note that this flag will be set or cleared based on change notifications according to [ERS02](#), [ERS03](#), and [ERS04](#).

- **<ERS07 – Equipment Recipe Store Service>**

The RMS shall provide a service to return to an authorized requestor the list of recipes known to be on a specified equipment plus related data, including which recipe components need to be added (see [ERS06](#)), which need to be removed from an equipment (see [ERS05](#)), and any unknown recipe components on the equipment (see [ERS12](#)).

- For example, this service would be used by a factory application responsible for doing the actual delete or download to the equipment as expressed by the related flags.
- It is expected that the service would allow the requestor to request all or only parts of this information (e.g., only the unknown recipe components found on the equipment).

- **<ERS08 – Request Equipment Recipe Component Delete>**

The RMS shall support a service to allow an authorized requestor to request the RMS to delete one or more recipe components from an equipment.

- Whether the delete is carried out actively by the RMS ([ERS10](#)) or passively ([ERS05](#)) is up to the RMS supplier and the factory system. This may vary from equipment to equipment within a factory. The RMS should be capable of both.

- **<ERS09 – Request Equipment Recipe Component Download>**

The RMS shall support a service to allow an authorized requestor to request the RMS to download one or more recipe components to an equipment.

- Whether the download is carried out actively by the RMS ([ERS11](#)) or passively ([ERS06](#)) is up to the RMS supplier and the factory system. This may vary from equipment to equipment within a factory. The RMS should be capable of both.

- **<ERS10 – Equipment Recipe Component Delete>**

The RMS shall be able to delete a recipe component from an equipment according to user-defined business rules or external instruction.

- Note that this request may need to be sent to an intermediary (such as the SECS/GEM Host), rather than directly to the equipment.

- **<ERS11 – Equipment Recipe Component Download>**

The RMS shall be able to initiate a recipe download to equipment according to user-defined business rules or external instruction.

- Note that this request may need to be sent to an intermediary (such as the SECS/GEM Host), rather than directly to the equipment.

- **<ERS12 – Unknown Equipment Recipe Component>**

The RMS shall be able to identify and flag any unknown recipe component on an equipment.

- This should be achieved by comparing the equipment recipe store list with the equipment components described in the recipe catalog.

## 6.5 Recipe Integrity

The factory needs to ensure that the correct equipment recipe, parameters, and settings are used on equipment for processing material. In today's factory, recipe integrity is commonly accomplished by downloading the correct recipe before each processing run. In the future, with the use of RaP, the need to download each time should be eliminated.<sup>7</sup>

---

<sup>7</sup> One exception is the case in which the recipe contains key intellectual property (IP) that must be protected. These may be used on the equipment and then immediately deleted.

- **<RI01 – Unique Identifier>**

The RMS shall assign each RMS recipe and recipe component a universally unique identifier (uuid according to ISO/IEC 11578:1996 ). The identity of the RMS recipe or recipe component shall be changed if the content of the recipe component(s) is changed.

- Such identifiers facilitate sharing of recipes between factories. RaP-compliant recipes are automatically assigned uuids as recipe identifiers.

- **<RI02 – Check Integrity>**

The RMS shall provide a service that will determine whether a supplied recipe component is the same as the copy of that recipe component in the factory recipe store.

- This service will be used for the “upload and compare” method of checking recipe integrity where the recipe component is uploaded from the equipment for comparison with the “golden copy” of the recipe component.
- This service may also be used for recipe components that contain equipment settings that the equipment knows as Equipment Constants and Status Variables. The SECS/GEM Host will be expected to put the equipment settings into the recipe component form for comparison.

- **<RI03 – Recipe Checksum>**

The RMS shall provide a service that will return the original checksum of a specified recipe component.

- One recipe integrity check is to ensure that the checksum of the recipe component on the equipment is the same as that in the factory recipe store.

## 6.6 Version Management

Recipe management is similar to software configuration management. As new versions of recipe components replace old ones, those that should be used in any given situation or processing context need to be managed and tracked. A uniform versioning method needs to be applied to recipes, recipe components, and collections of recipes. This method should also allow for tracking the recipe history.

- **<VM01 – Versioning Recipes >**

The RMS shall support versioning of RMS recipes.

- A “version” of an RMS recipe includes the specification of the version of all associated recipe components as well as the values of all data associated with the recipe definition. Fixed parameter sets (see [PM01](#)), where used, are a part of the versioned recipe.

- **<VM02 – Versioning Recipe Components >**

The RMS shall support versioning of recipe components.

- **<VM03 – Version History>**

The RMS shall maintain the version history of RMS recipes and recipe components.

- The version history includes the identifiers of antecedent recipes, the date each version was created, who made the change, and the reason for the change.

## 6.7 Change Control

To ensure that changes to RMS recipes and recipe components are introduced only at the proper time and only after appropriate testing, the factory needs to have a robust change control methodology in place. The change control methodology should provide flexibility and accountability.

- **<CC01 – Change Control Sign-off>**

The RMS shall provide a change control mechanism with appropriate approval authorization levels and sign-off loops.

- Change control covers recipes and recipe components.
- An approver must be able to approve/sign-off an arbitrary set of recipes at one time.

- **<CC02 - Change History>**

The RMS shall maintain a change history of each version of an RMS recipe or recipe component.

- The change history includes what status change occurred, the reason for the change, who made change, approver, date approver signed-off or rejected and if rejected, the reject reason.

- **<CC03 – Recipe Check-in/Check-out>**

The RMS shall provide check-in/check-out services to control modifications of recipes and recipe components and to reserve them for selected activities (e.g., the approval process).

## 6.8 Authentication and Authorization

Factory systems need to provide security to prevent unwarranted access or changes to proprietary information. Systems also need to ensure that only authorized personnel/applications have access to information and that only authorized personnel/applications are allowed to make changes to information.

- **<AA01 – Authentication>**

The RMS shall provide a mechanism to properly authenticate any user or application requesting access to the RMS system.

- The RMS must be able to support universal authentication (e.g., LDAP).

- **<AA02 – Authorization>**

The RMS shall provide a mechanism to ensure RMS functions are performed only by properly approved and qualified users and applications.

- The list of privileges and groupings should be customizable by the user. However, the list should include all services and functions listed in this requirements set.
- It should be possible to define authorization based upon such RMS recipe attributes as tool type, product, or route.

## 6.9 Parameter Management

The need for recipe parameters is continuing to increase as process control techniques and implementations proliferate and as the goal to reuse recipes among equipment and processes is realized. The RMS must address the increasing complexities that results as parameter usage increases and multiple systems attempt to influence the setting of the same parameters.

- **<PM01 – Parameter Sets>**

The RMS recipe definition shall include the specification of the recipe parameters that can be supplied at run time.

- The parameter definition includes type, limits, tolerance, and default value.

- **<PM02 – Parameter Value Sets>**

As needed by the user, the RMS shall manage as a recipe component the set of parameter values to be used with a specific recipe.

- Fixed sets of parameter values can be used to facilitate shared recipe components (see Section 6.10).

- **<PM03 – Parameter Rules>**

The RMS shall store rules dealing with how parameter values are determined, merged, and set and limits enforced.

- These rules determine which applications can supply parameter values. Note that there may also be fixed sets of parameters stored as recipe components.
- Where multiple applications supply parameter values, these must be arbitrated (merged or selected).
- Limits normally define absolute extremes for a value. However, another example of a limit is how much a parameter can be adjusted at any one time (rate of change).

- **<PM04 – Parameter Rules Access>**

The RMS shall provide a service to return the set of parameter rules.

- Enforcement/execution of the parameter rules may be done by an external application.

## 6.10 Shared Recipe Components

Two recipes may use the same recipe component, yet achieve different results based on parameter values used or, if a multi-part recipe, based upon the other recipe components used.

Factories typically store and manage many thousands of recipe components. In many cases, sets of similar recipe components may differ by a few parameter settings. The total number of recipe components can be reduced significantly if those differences can be factored out as parameter settings. The RMS needs to facilitate this process (i.e., recipe component sharing).

- **<SR01 – Recipe Component Sharing>**

The RMS shall allow recipe components to be used by multiple RMS recipe definitions.

- **<SR02 – Facilitate Shareable Recipe Components>**

The RMS shall provide the user with assistance in identifying similar recipe components.

- The RMS may compare recipe components for a particular equipment and list those sets that differ by only a small amount.

- **<SR03 – Shared Recipe Change Notice>**

When a shared recipe component is updated and a new version of that recipe component is created, the RMS shall notify the user about any RMS recipes that reference that RMS component in case they should be updated to reference the new version.

## 6.11 Report Generation

Factory personnel need reports on manufacturing activity. The RMS needs to be able to produce the appropriate reports in a timely manner.

- **<RG01 – Report Generation>**

The RMS shall provide a service to generate a preconfigured (canned) set of reports on the contents and activities of the RMS.

- **<RG02 – Custom Report>**

The RMS shall provide a service to allow a user-customizable report to be generated.

- **<RG03 – Auto-generation>**

The RMS shall provide a mechanism to schedule the automatic generation of reports at user-designated times.

## 6.12 Recipe Component Viewing and Modification

Manufacturing and engineering personnel need the flexibility of reviewing and creating recipe components outside the fab environment. Today, most recipe components have a format that is specific or proprietary to an equipment supplier. However, the trend is toward more open formats that are accessible to the factory user. The degree to which these components may be edited varies. The supplier should provide whatever editing and viewing capability that is feasible.

- **<REV01 – Recipe Component Viewer>**

The RMS shall provide a service to view recipe component content.

- If the format of the recipe component is known, it should show the contents in an intelligible form. If it is not known, then even a hexadecimal view is useful.
- The equipment supplier may be able to supply the recipe component format or a decoder application to assist with this capability.
- XML format is expected to come into more common use for recipe components. The RaP standard (Section 2.2) also requires an XML-formatted header that includes descriptive information about the recipe component.

- **<REV02 – Recipe Component Generator>**

The RMS shall provide a service to create new recipe components or modifications based on formulas or instructions on how to make the modification.

- The recipe generation will typically involve small programmatic changes to an existing recipe to change recipe settings (e.g., for process control) or recipe identification (e.g., for uniform factory naming conventions).

- **<REV03 – Universal Recipe Component Editor>**

The RMS shall provide a capability to edit any recipe component for which the format, syntax, and content rules are known.

- This universal editor needs to be accessible on an engineer’s desktop computer for ease of use.

### 6.13 Requirements Table

Table 1 lists the common RMS requirements by category. In addition to the Requirement ID and the Requirement Name, each entry includes the two classification columns. In the Core/Active column, “C” is used for Core and “A” is used for Active. A core requirement is central to an RMS. The Core capabilities do not require that the RMS initiate communications to other factory entities and are thus considered “passive.” “Active” requirements/capabilities do require that the RMS initiate external communications. These are required in the factory system, but today are not universally considered part of the RMS. A commercial RMS supplier should be prepared to meet these active requirements as needed by their customers.

In the Present/Future column, “P” is used for Present and “F” is used for Future. Present requirements are addressed by existing RMS applications in factories today. Most present requirements listed in this document are expected to be needed even as future requirements evolve. Future requirements are expected to be in general use within 3–5 years. In many cases, they are already used in some RMS implementations. These future requirements are often supported today by other factory applications, but are expected to migrate into the RMS of the future.

**Table 1 Requirements Table**

Requirement ID	Requirement Name	Core/Active	Present/Future
<b>General RMS Requirements</b>			
<a href="#">GR01</a>	User Interface Access	C	P
<b>Recipe Catalog</b>			
<a href="#">RC01</a>	Recipe Definition	C	P
<a href="#">RC02</a>	Recipe Creation	C	P
<a href="#">RC03</a>	Get Recipe Manifest	C	P
<a href="#">RC04</a>	Recipe Access	C	P
<a href="#">RC05</a>	Recipe Relationships	C	P
<a href="#">RC06</a>	Recipe Descriptive Data	C	P
<a href="#">RC07</a>	Recipe Component Descriptive Data	C	P
<a href="#">RC08</a>	Recipe Lifecycle Status	C	P
<a href="#">RC09</a>	Archive Recipe	C	P
<a href="#">RC10</a>	Recipe Actions	C	P

Requirement ID	Requirement Name	Core/Active	Present/Future
<a href="#">RC11</a>	Get Recipe List	C	P
<a href="#">RC12</a>	Recipe Browser	C	P
<b>Factory Recipe Store</b>			
<a href="#">FRS01</a>	Store Golden Copy	C	P
<a href="#">FRS02</a>	Recipe Component Access	C	P
<a href="#">FRS03</a>	Recipe Component Upload	C	P
<a href="#">FRS04</a>	Get Recipe Component List	C	P
<a href="#">FRS05</a>	Archive Recipe Component	C	P
<a href="#">FRS06</a>	Recipe Component	C	P
<a href="#">FRS07</a>	Delete Component	C	P
<a href="#">FRS08</a>	Never Replace Recipe Component	C	F
<a href="#">FRS09</a>	Export/Import Recipe Components	C	P
<b>Equipment Recipe Store</b>			
<a href="#">ERS01</a>	Equipment Recipe Store List	C	P
<a href="#">ERS02</a>	Equipment Recipe Change Event	C	F
<a href="#">ERS03</a>	Equipment Recipe Store List Update	C	F
<a href="#">ERS04</a>	Equipment Recipe List Request	A	F
<a href="#">ERS05</a>	Equipment Recipe Component Delete Flag	C	F
<a href="#">ERS06</a>	Equipment Recipe Component Download Flag	C	P
<a href="#">ERS07</a>	Equipment Recipe Store Service	C	F
<a href="#">ERS08</a>	Request Equipment Recipe Component Delete	C	F
<a href="#">ERS09</a>	Request Equipment Recipe Component Download	C	F
<a href="#">ERS10</a>	Equipment Recipe Component Delete	A	F
<a href="#">ERS11</a>	Equipment Recipe Component Download	A	F
<a href="#">ERS12</a>	Unknown Equipment Recipe Component	C	P
<b>Recipe Integrity</b>			
<a href="#">RI01</a>	Unique Identifier	C	P
<a href="#">RI02</a>	Check Integrity	C	P
<a href="#">RI03</a>	Recipe Checksum	C	P
<b>Version Management</b>			
<a href="#">VM01</a>	Versioning Recipes	C	P
<a href="#">VM02</a>	Versioning Recipe Components	C	P
<a href="#">VM03</a>	Version History	C	P
<b>Change Control</b>			
<a href="#">CC01</a>	Change Control Sign-off	C	P
<a href="#">CC02</a>	Change History	C	P
<a href="#">CC03</a>	Recipe Check-in/Check-out	C	P
<b>Authentication and Authorization</b>			
<a href="#">AA01</a>	Authentication	C	P
<a href="#">AA02</a>	Authorization	C	P
<b>Parameter Management</b>			
<a href="#">PM01</a>	Parameter Sets	C	P
<a href="#">PM02</a>	Parameter Value Sets	C	P
<a href="#">PM03</a>	Parameter Rules	C	F
<a href="#">PM04</a>	Parameter Rules Access	C	F

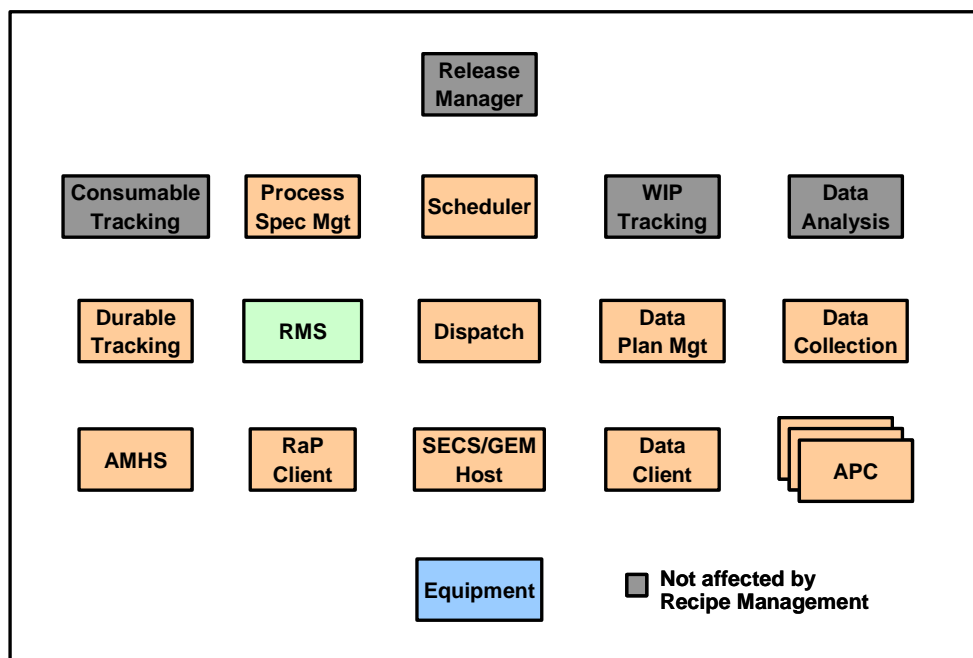
Requirement ID	Requirement Name	Core/Active	Present/Future
<b>Shared Recipe Components</b>			
<a href="#">SR01</a>	Recipe Component Sharing	C	F
<a href="#">SR02</a>	Facilitate Shareable Recipe Components	C	F
<a href="#">SR03</a>	Share Recipe Change Notice	C	F
<b>Report Generation</b>			
<a href="#">RG01</a>	Report Generation	C	P
<a href="#">RG02</a>	Custom Report	C	P
<a href="#">RG03</a>	Auto-generation	A	F
<b>Recipe Component Viewing and Modification</b>			
<a href="#">REV01</a>	Recipe Component Viewer	C	P
<a href="#">REV02</a>	Recipe Component Generator	C	F
<a href="#">REV03</a>	Universal Recipe Component Editor	A	F

## 7 RECIPE MANAGEMENT EFFECTS ON THE FICS

The effects of recipe management on the factory system are not limited to the RMS and SECS/GEM Host. This section looks at these influences on functional areas typically found in factory systems.

There is no universally accepted partitioning of semiconductor factory systems. The functional areas used here are not intended to represent any particular factory system. Readers are encouraged to map these functional areas appropriately to their own factory system partitions.

The set of functional areas is not intended to be comprehensive. Only those areas significantly affected by recipe management are discussed (see Figure 3).



**Figure 3** Factory Components

## 7.1 Scheduling

Scheduling is not typically involved in current recipe management approaches. However, there may be a future relationship.

The trend toward larger more complex equipment recipes combines with a trend toward smaller lot sizes to threaten large increases in bandwidth needed for recipe transfer and in-transfer time. Today many recipes are downloaded to the equipment each time they are used. In the future, this practice may be eliminated if the factory can trust that a recipe downloaded to an equipment will remain, unchanged, on that equipment for any future use. RaP supports this approach by ensuring recipe integrity and by providing for recipe parameterization where parameter values supplied at run time can replace small on-going adjustments to the recipe.

When the recipe does need to be downloaded to the equipment, it is recommended it be downloaded before the need is immediate. Some factory application must periodically review the schedule (e.g., per shift) to predict what material might be processed at each equipment. The recipes components needed for these process jobs must be compared to the recipe components currently resident on the equipment.

This “look ahead” approach should also interact with the SECS/GEM Host (or scheduler or dispatcher) to learn when the equipment is idle so that recipe download can occur at this time. In this way, the factory can avoid the risk that recipe download will affect either processing or the related command and control messaging.

## 7.2 Dispatching

The effect of recipe management on dispatching will be small in most cases. Most dispatchers operate on a high level, akin to “process lot x at equipment y.” It is typical that the dispatcher has some rules and prerequisites that determine when to dispatch a lot to an equipment. A factory might choose to require that the needed equipment recipe be resident on the tool before it can dispatch the lot. Whether such rules exist will determine if a relationship to recipe management exists.

Recipes may be designed for specific processing chambers or specific kinds of processing chambers (model/revision/etc.). Another rule for dispatching might be that the processing chamber, as specified by the recipe, must be available and functional.

## 7.3 Process Specification Management

The Process Specification, or “flow,” defines the route that substrates may take through the factory and the details of how processing should be accomplished at each step. Some areas of recipe management may overlap with process specification management:

- The identification of the RMS recipe to be used for a given step might be specified in the process specification.
  - For multi-part recipes, the RMS is typically needed to link all the needed recipe components to that single identifier in the process specification.
  - Alternatively, the RMS may identify the recipe based on context.
- In some cases, sets of recipes for a process might need to be changed as a group. This can be done as a revision of the process specification more easily than in the RMS.

#### **7.4 Durable Tracking**

When durables are involved in processing, the recipe may specify the durable to be used. For example, a stepper recipe may be designed to work only with a specific reticle. The factory system must understand any such recipe limitations. The specification of the durable to be used may also be supplied as a recipe parameter.

#### **7.5 Equipment Maintenance Tracking**

Recipe components may be designated to run only on specific processing chambers. The factory system must understand any such recipe limitations.

#### **7.6 Data Collection Plan Management**

Data collection plans are specifications and have management requirements that are similar to recipes. It is reasonable to consider whether these two specification management type systems can be combined.

Data collection plans and recipes may be related. An RMS recipe may require that a specific data collection plan be used. For example, a high difficulty product might require significantly more data collection than older technology products.

#### **7.7 Data Collection**

Equipment data collection must be designed so that recipe and parameter usage traceability information is collected and stored for later analysis, as needed. As processing becomes more complex, with multi-part recipes and parameter values that change from run to run (or wafer to wafer), it is important to enable traceability of which recipe component and parameter values are used in every processing instance.

For each wafer (or group) that is processed, an event report should be available when a recipe component begins execution on that wafer/group. Consequently for each wafer/group, there might be an event for “sequence” recipe start and then one more for each “chamber” recipe started on a multi-chamber tool. Each event would include the identifier of the recipe component and the list of parameters/values supplied to that recipe component.

#### **7.8 SECS/GEM Host**

The scope of SECS/GEM Host implementations varies widely. These implementations also vary in terms of the other factory components with which they interact. Recipe-related capabilities that SECS/GEM Host applications may possess include the following:

- Process control settings must be retrieved as needed before (and perhaps during) process execution. The SECS/GEM Host may determine which parameter settings are needed and request them (or use default values where appropriate). In today’s systems, there is typically only one source for all settings (run-to-run control). Future advances may require a more flexible approach that can handle multiple sources of settings.
- The SECS/GEM Host will typically check that any needed recipe files are on the tool and download any that are missing.
- Data collection historically passes through the SECS/GEM Host. The SECS/GEM Host also typically configures event reporting (including recipe traceability data). This is

expected to change with the advent of Equipment Data Acquisition (EDA), which provides for separate XML/SOAP-based data clients.

- The SECS/GEM Host manages equipment processing (e.g., control job and process job lifecycle). In this process, the SECS/GEM Host specifies the equipment recipe to use and the recipe parameter values.
- The SECS/GEM Host performs all recipe transfer, validation, presence checking, and related services. This may change in the future when RaP clients may use the XML/SOAP interface<sup>8</sup> to connect to the equipment.

## 7.9 Process Control

A process control system must handle different control models for different products. It must be told which recipe is being run and/or what product is being run. It must also know what recipe parameters must be set and their limits. Rules for how to use and manipulate the parameters may reside in the RMS.

## 8 FUTURE VISION FOR FACTORY RECIPE MANAGEMENT

### 8.1 Trends That Will Affect Recipe Management

Advances in semiconductor manufacturing have added significant complexity to the production process and new trends promise even more. Where single chamber, single function equipment was once universal in the factory, multi-chamber tools and processing chambers where multiple operations can be performed on the wafer are common. Periodic manual equipment tuning has given way to constant monitoring and adjustment of equipment. Where, in the past, a single recipe specification would suffice to process a 25-wafer lot, now different recipes are being used for each wafer.

Changes in the factory system are commonly driven by either advancements in the devices being manufactured or the need for improvements in processing efficiency. These types of changes are represented here as “Feature Shrink” and “Next Generation Processing,” respectively. Device manufacturers today are concurrently driving these changes to extreme limits. This pressure on current factory systems may produce significant changes in factory control, specifically to recipe management systems.

#### 8.1.1 Feature Shrink

Process technologists are creating ever smaller features. As each new technology node is encountered, new challenges arise for controlling the manufacturing equipment to ever tighter tolerances. Approaches to achieving this new level of control have multiplied in recent years.

Traditionally, equipment is “tuned” on a regular basis by an equipment technician. This tuning process results in a periodic change in the equipment performance baseline, as the tuning process attempts to return the drifting equipment performance to its optimum state.

---

<sup>8</sup> Note that the XML/SOAP interfaces for EDA and RaP are both based on SEMI E132. Once one is available, there is little technological barrier to the other.

Today's factories also have run-to-run (R2R) process control on critical process tools. R2R control makes lot-by-lot adjustments based on past equipment performance by either changing the processing recipes or using recipe parameters to modify the execution of the recipe. The R2R control application must be aware of any "tuning" activity, since it is trying to adjust for the processing drift that tuning is meant to eliminate. A new trend in R2R control is a move toward making processing adjustments more frequently, first on a wafer-by-wafer basis and later perhaps during wafer processing.

Recently, new control techniques are beginning to coexist or replace run-to-run control. For example, "Virtual Metrology" uses large amounts of processing data to model and predict processing outcome. This technique also generates processing adjustments that affect recipe parameters. Factories are also beginning to feed-forward process control, i.e., using information about the state of the substrate to suggest process adjustments. A simplistic example is adjusting for underdeveloped film by over-etching the wafer.

Machine-to-machine matching (also called chamber matching) is a technique for adjusting equipment so that processing results are essentially identical. Adjustments may be made by technicians (e.g., calibration/tuning) or by the SECS/GEM Host supplying recipe parameter offsets.

Another trend is the move toward actively monitoring the equipment state in addition to the process results. Predictive and preventive maintenance (PPM) is one name for this approach, which gathers large amounts of detailed equipment data that can allow an application to predict and adjust for the wear level of equipment components as well as detecting key signs of impending failure of other components. Again, equipment adjustments result from this technique.

The equipment recipe is responsible for specifying the processing conditions for a step in the process. When other entities change these processing conditions, the changes must be coordinated in a predictable way. The recipe management system is responsible for the equipment recipe as well as specifications for what run-time changes can be made and the rules for coordination when multiple entities are contributing changes. When there is no process control or only a single entity makes such adjustments, there typically is no problem. However, as more applications attempt to "help" improve processing, such coordination will become critical to the success of the overall processing goal.

### **8.1.2 Next Generation Processing**

The industry is currently building consensus to move past the current 300 mm processing approach into the next generation of processing techniques. This includes an anticipated wafer size increase to 450 mm as well as evolutionary changes to factory systems for both 300 mm and 450 mm to promote new levels of processing efficiency.

The goals for the evolution of 300 mm factories (i.e., 300 Prime) include reduced cycle time and die cost through reduced lot size, elimination of non-value-add activities, and improved equipment availability. The move to 450 mm builds on 300 Prime by continuing lowering costs as problems created by the larger wafer size are addressed.

The move to smaller lot size affects the amount of time available for processing setup without delaying process start. Most process equipment process one lot at a time with a second lot starting only after the last wafer of the previous lot has moved into the equipment. This provides a lead time equal to approximately 24 wafers x the time of the first step within that equipment.

This can amount to several minutes. Assuming a piece of equipment with two loadports, during this lead time the previous lot must be replaced on the equipment with the next lot, the recipe components for the new lot must be downloaded and the equipment must be instructed to process the lot.<sup>9</sup>

As the number of wafers in a lot is reduced, the lead time is shortened. Equipment recipe downloads contribute an increasing share of this setup effort as recipe size increases and traffic on the SECS/GEM communication link becomes more congested with expanding data reporting.<sup>10</sup> The effect on the RMS is pressure to eliminate recipe download at setup time. This means that recipes must be downloaded in advance and the integrity of those recipes must be guaranteed against unauthorized modification.

Another effect on recipe management is the continued increase in the complexity of individual equipment as more have multiple process chambers (and multi-part recipes) and offer recipe parameters for process adjustments at setup time or even during processing. The RMS must increasingly understand the interrelationships among recipe components for multi-part recipes and deal with the more complex recipe parameterization schemes needed for multi-part recipes.

As recipes and recipe execution becomes more complex, it becomes more important than ever to capture exactly which recipe component is used in each equipment step for each wafer (or wafer group) and which recipe parameter values affected this processing instance. This is termed “recipe traceability.” Some examples of where traceability is particularly important are as follows:

- Recipe parameter values are passed to the equipment before the processing begins, but the equipment allows updates for the values during processing. A “race condition” may exist where the new value may or may not be used for a particular wafer.
- Equipment recipe sequences may be allowed to branch (e.g., to choose from equivalent process modules). The choice of paths at the branch will often result in the execution of a different recipe component or may require different recipe parameter settings where the same recipe can be used on similar process modules.

The reporting of traceability information (recipe identifiers and parameter values used) in each case provides the factory with the ability to fully investigate processing problems and differentiate between poor processing performance and incorrect instructions.

## 8.2 RMS in Transition

A goal of this document is to guide the industry toward a common, robust set of future requirements that raises the quality level for all device makers while creating efficiencies that will lessen the cost impact of transition to RMS. When the requirements for a factory component (hardware or software) are the same (or nearly the same) from all customers, a supplier must create the solution only once to meet everyone’s needs.

When the current and future requirements are compared, no obvious conflicts between the two sets are apparent. This indicates that implementers of recipe management systems with a forward-looking overall system design can begin by meeting the current RMS requirements and

---

<sup>9</sup> This is a simplification. There are additional, often needed setup activities not mentioned here.

<sup>10</sup> The Equipment Data Acquisition (EDA) interface promises to lessen the later effect by moving data traffic to a different communication link; however, load on the equipment controller may still have an effect.

then add the features defined in the future requirements in an evolutionary process according to customer priorities.

One area that is not uniformly addressed by RMS today is the set of “active” features. In most cases, the RMS today is a passive application. It stores recipes. It maintains recipe descriptions. It enforces logic in changing the contained data and status. However, it does not typically act upon the information it has. It does not request equipment directory lists to update the recipe store list. It does not transfer recipes to/from the equipment. To truly manage the set of recipes in the factory, such active features are required. In the future, the RMS is expected to provide this capability.

Transition to the “future” RMS requirements depends, in part, upon the availability of equipment support for these requirements. In particular, the migration of production equipment to support SEMI E139 (the RaP standard) will be needed before the RMS can meet all the “future” requirements listed in this document.

A transition period is anticipated as RaP support expands across the equipment supplier base. This transition is expected to begin in 2008 and may extend over 2–3 years. During this time, factory recipe management systems will be expected to apply the “present” RMS requirements to the older equipment while meeting “future” requirements for RaP-enabled equipment.<sup>11</sup>

## 9 CONCLUSION

As the needs of the factory increase and the capabilities of the equipment expand, factory recipe management systems must take advantage of the resulting opportunity to add value to the factory. This value is realized in several areas, including the following:

- Single point of operational control for all recipes in the factory
  - Central control all of the elements of an RMS recipe
  - Improved opportunity to ensure correct recipe usage
  - Ability to deal with recipes from across the entire product flow in a coordinated way
- Better protection of the intellectual property represented by equipment recipes
- Application of business rules uniformly across the factory’s recipes
- Single point of integration with other factory systems
- Active version control of recipes to provide finer control of what recipe is used with each process context
- Management of recipe parameter sets to facilitate equipment matching (improved processing control) and recipe sharing (reduction of recipes)
- Promotion of off-tool recipe editing (saving tool time)

This set of RMS requirements is a powerful tool intended to move ISMI members toward the efficient realization of its potential value.

---

<sup>11</sup> Some “future” requirements do not require RaP-enabled equipment. Application of such future requirements to the current equipment base will be a differentiator among RMS implementations.

## Appendix A – Member Survey Responses Summary

ISMI member companies were invited to participate in teleconferences where this work was discussed. The members were also requested to complete a survey to identify current and future recipe management system requirements. Six members returned completed surveys. Their responses are summarized in Table A-1. The summary was reviewed and discussed by the RaP Working Group. Individual member responses and the identity of the participating members are not publicly available from ISMI.

**Table A-1 Member Survey Responses Summary**

Does your fabs' RMS support the following? If not, will you need it in future?	Present Version	Future Version	Not at All	Comments
<b>Central Storage</b>				
Golden copy	Yes(6)			
Recipe access from equipment, factory applications, remote editors	Yes(5)	Yes(1)		
Recipe upload / check-in	Yes(6)			
Recipe Status	Yes(6)			
Equipment recipe store tracking & management	Yes(4)	Yes(1)		One member no response
<b>Version Management</b>				
Version history	Yes(6)			Whole recipe version stored with date range indicating which dates it was valid. Would like to include which tool it came from and when it was uploaded. Archiving would be nice to have based on (possibly) date. Archive management.
Manage a set of recipes – approve together	Yes(4)	Yes(2)		
<b>Change Control</b>				
Approval levels and signoff (using existing authentication systems)	Yes(6)			
Change history	Yes(6)			
User privilege control, including applications and equipment	Yes(4)	Yes(1)		One member no response
<b>Recipe Integrity / Verification / Validation</b>				
Ensure correct recipe usage.....If so how?	Yes(4)	Yes(1)		One member no response
Upload and compare equipment constants (ECID's) from equipment	Yes(2)	Yes(4)		
Upload and compare Status variables (SVID's) from equipment	Yes(2)	Yes(2)	No(2)	
Check recipe for correct structure	Yes(3)	Yes(2)	No(1)	
Check recipe for correct syntax	Yes(1)	Yes(2)	No(3)	

Does your fabs' RMS support the following? If not, will you need it in future?	Present Version	Future Version	Not at All	Comments
<b>Parameter Management</b>				
Which parameters can be changed by which applications / people	Yes(4)	Yes(2)		
Equipment constants – like parameters	Yes(3)	Yes(3)		
Manage the sets of parameter values and sources of values	Yes(4)	Yes(2)		
Store tolerance for tuning parameters / constants	Yes(4)	Yes(2)		
Parameters values compared with upper and lower tolerances	Yes(4)	Yes(2)		
Parameter change arbitration / control / merging	Yes(3)	Yes(3)		
Parameter setting by recipe modification per job	Yes(3)	Yes(2)	No(1)	
<b>Share Recipes</b>				
Among different equipment	Yes(5)	Yes(1)		
Among different processes	Yes(3)	Yes(2)	No(1)	
Report generation	Yes(2)	Yes(2)	No(1)	One member no response
<b>Recipe Editor &amp; Viewer</b>				
Recipe viewer / decoder	Yes(5)	Yes(1)		
Recipe generator – new recipes / versions based on formulas or instructions	Yes(2)	Yes(3)	No(1)	
Remote editor interfaces	Yes(1)	Yes(3)	No(2)	
<b>Universal Recipe Editing</b>				
Remote / off-equipment edit	Yes(3)	Yes(2)	No(1)	
Edit all recipes (formatted, unformatted, binary)	Yes(2)	Yes(2)	No(1)	One member no response
Support recipe templates	Yes(1)	Yes(2)	No(1)	Two members no response
What additional recipe management features do you need in the future?	(1) Allow multiple recipes to point at common sub recipes (2) Recipe comparison features without decoders (3) Recipe capability check with tool software rev's. (4) Once recipes are downloaded for a particular process job, then it is only viable for that job and does not persist to tool level. (5) Recipe type control at a group level. (6) We need RaP so we can safely avoid downloads.			
What are the primary drivers of these new RMS features?	(1) Effective and efficient RM. (2) Fab to Fab alignment. (3) Small-lot manufacturing			
Do you have plans to request / use RaP (Recipe and Adjustable Parameter) (SEMI Standard E139)?	Yes(3)	Yes(1)		
If so, what is the timing for RaP usage?	(1) One year (2) Upon equipment supplier implementations (3)No exact planning about this at the moment.			
Are there requirements you cannot	Yes(2)	Yes(1)		

Does your fabs' RMS support the following? If not, will you need it in future?	Present Version	Future Version	Not at All	Comments
get / implement (or are difficult) with today's equipment or recipes?				<p>(1) Equipment that needs to upgrade recipes with new tool software is problematic due to coordination of software upgrades and different recipe versions running.</p> <p>(2) Tool side translators that determine recipe parameter context on the tool at runtime: makes it hard for offline editing and recipe sharing. (e.g. gas1, gas2...)</p> <p>(3) Some tools do not allow recipe download when in bad state (e.g. chamber down). Would like notification of error and description.</p> <p>(4) Validation of recipe (Some Equipments store timestamps inside, some do not upload full recipes);</p> <p>(5) Parameter Compare (Format not know for all recipes)</p> <p>(6) Upload of sequence and parameter recipes in the same way without using different SECS messages and data.</p>

## Appendix B – Supplier Investigation

### B.1 Recipe Management Systems Suppliers

Potential Recipe Management System (RMS) suppliers were identified from candidates submitted from members, candidates known from other avenues and from a search on the internet. A total of 12 potential candidates were identified.

From investigations on company websites and discussions with people in the industry several products/companies were eliminated. One was a consulting firm with no RMS product. Another was a custom software and service company with no RMS product. A third was a component of an MES and had limited recipe management functionality. Eliminating these companies reduced the list to nine viable candidates. These companies were contacted and requested to supply information. Five responded and agreed to fill out a survey. Four of those five agreed to be listed in this document:

1. RecipeWorks from Adventa Control Technologies, Inc.  
website: <http://www.adventact.com/products/recipeworks/index.htm>
2. eRMS from BISTel  
website: [http://www.bistel-inc.com/eng/rms\\_eng.asp](http://www.bistel-inc.com/eng/rms_eng.asp)
3. LineWorks Recipe Management <sup>TM</sup> from camLine  
website: <http://www.camline.com/>
4. RME from Synopsis  
website: [http://www.synopsys.com/products/mym/rme\\_ds.html](http://www.synopsys.com/products/mym/rme_ds.html)

These companies were sent a survey, created by the ISMI RaP Working Group, consisting of a series of questions about RMS requirements. All companies returned completed surveys. A telephone interview was conducted with each company to review the survey for completeness and to ensure the questions were well understood.

Since this activity was to gather requirements and not an evaluation of the products, the suppliers were taken at their word about features and functionality. All RMSs did appear to have basic functionality.

A summary of the survey responses is shown in Table B-1. The individual company survey responses are not publicly available from ISMI.

**Table B-1 Survey Responses**

Does your recipe management product support the following?	Present Version	Future Version	Not At All	Comments
<b>Central Storage</b>				
Golden copy	Yes(5)			
Recipe access from equipment, factory applications, remote editors	Yes(5)			
Recipe upload / check-in	Yes(5)			
Recipe Status	Yes(5)			
Equipment recipe store tracking & management	Yes(4)		No(1)	(1) RMS provides the ability to persist and query this information, however as it is a passive solution; it does not directly fetch this information off the tool, but instead relies on the EI layer feeding it this information.
<b>Version Management</b>				
Version history	Yes(5)			
Manage a set of recipes – approve together	Yes(3)	Yes(2)		
<b>Change Control</b>				
Approval levels and signoff (using existing authentication systems)	Yes(4)	Yes(1)		
Change history	Yes(5)			
User privilege control, including applications and equipment	Yes(5)			
<b>Recipe Integrity / Verification / Validation</b>				
Ensure correct recipe usage ... how?	Yes(5)			
Upload and compare equipment constants (ECID's) from equipment	Yes(2)	Yes(1)	No(2)	(1) This data is stored in the RMS system, but it is the EI layer that typically uses that info to check if the ECID's are set correctly on the tool before using the recipe.
Upload and compare Status variables (SVID's) from equipment	Yes(1)	Yes(2)	No(2)	
Check recipe for correct structure	Yes(3)		No(1)	
Check recipe for correct syntax	Yes(3)		No(1)	
<b>Parameter Management</b>				
Which parameters can be changed by which applications / people	Yes(2)		No(3)	
Equipment constants – like parameters	Yes(3)		No(1)	
Manage the sets of parameter values and sources of values	Yes(5)			(1) Parameters are user configured – data type, bounds, default values. Source can come from API arguments at runtime, default values or directly from integrated R2R control system

Does your recipe management product support the following?	Present Version	Future Version	Not At All	Comments
Store tolerance for tuning parameters / constants	Yes(4)		No(1)	(1)Upper and lower bounds are configurable in RMS, tolerance is managed in R2R control system
Parameters values compared with upper and lower tolerances	Yes(5)			(1) Runtime check is performed to validate bounds
Parameter change arbitration / control / merging		Yes(3)	No(2)	(1) Parameters are merged at runtime into body of recipe during download API – arbitration is done in our R2R control system
Parameter setting by recipe modification per job	Yes(4)		No(1)	
<b>Share Recipes</b>				
Among different equipment	Yes(5)			
Among different processes	Yes(4)	Yes(1)		
Report generation	Yes(3)	Yes(1)	No(1)	
<b>Recipe Editor &amp; Viewer</b>				
Recipe viewer / decoder	Yes(4)		No(1)	
Recipe generator – new recipes / versions based on formulas or instructions	Yes(3)	Yes(2)		
Remote editor interfaces	Yes(2)	Yes(1)	No(1)	
<b>Universal Recipe Editing</b>				
Remote / off-equipment edit	Yes(3)	Yes(1)	No(1)	(1) Would like general standard for invoking tool vendor's recipe editors off of tool, or some generic recipe editing framework so recipes can be edited off-tool
Edit all recipes (formatted, unformatted, binary)	Yes(3)	Yes(1)	No(1)	
Support recipe templates	Yes(3)	Yes(1)		
What additional recipe management features do you plan to support in the future?	Current Features- Recipe comparison (between equipments, chambers, versions)- No Limitation on number of recipe version- Recipe Reuse: Use old recipe version- Security: Able to block screen capture, print, export features for added security- Recipe delete in equipment- Export / import recipe data to EXCEL format- Recipe quick re-validation: Check if important parameters have limits defined.- Workflow: Customize validation scenario with dynamic deployment capabilityFuture Plans - Recipe parameter value drift compensation- Parameter correlation: Correlated parameters must be changed when a parameter value changes- Integration to R2R, FDC, EPT, PMM -Control Job / Process Job editing / uploading / downloading through RMS is currently being implemented and tested at a customer site (request from customers)			
What are the primary drivers of these new RMS features?	(1) The primary drivers for new features are customer driven.			
Do you have plans to support RaP (Recipe and Adjustable Parameter) (SEMI Standard E139)?	Yes(2)			

Does your recipe management product support the following?	Present Version	Future Version	Not At All	Comments
If so, what is the timing for RaP support?				<p>(1) On an "customer-need" basis only</p> <p>(2) Our current plans to support RaP is as follows:</p> <ul style="list-style-type: none"> <li>- Beta Version Release: 2007 / Q4</li> <li>- Site Test Version Release: 2008 / Q1</li> <li>- Complete Version Release: 2008 / Q3</li> </ul> <p>Currently the customers are not specifically asking for RaP support.</p> <p>(3) TBD for complete RaP support. We already support recipe export in XML as well as multi-body editing in our system</p> <p>(4) 2008</p>
Are there requirements you can not meet (or have difficulty meeting) with today's equipment or recipes?				<p>(1) Offline editing is biggest problem. Would like an interface, perhaps, to equipment recipe editors to allow for offline editing that looks like the tools editors</p> <p>(2) One of key challenges faced in terms of provided recipe management system lies in what the physical equipment interface provides. Many tool manufacturers do not expose the underlying recipe structure in detail so that we do the following effectively:</p> <ol style="list-style-type: none"> <li>1. Developing an external recipe editing capability</li> <li>2. Being able to specific sections of a recipe</li> <li>3. Expose specific critical parameters for a given recipe</li> </ol>



**International SEMATECH Manufacturing Initiative  
Technology Transfer  
2706 Montopolis Drive  
Austin, TX 78741**

**<http://ismi.sematech.org>  
e-mail: [info@sematech.org](mailto:info@sematech.org)**