Increasing inspection equipment productivity by utilizing Factory Automation SW on TeraScan 5XX systems

Thomas Jakubski*[a], Michal Piechocinski[a], Raphael Moses[a], Bharathi Bugata[b], Heiko Schmalfuss[b], Ines Köhler[c], Jan Lisowski[c], Jens Klobes[c], Robert Fenske[c]

[a]Advanced Mask Technologies Center, Raehnitzer Allee 9, 01109 Dresden, Germany
[b]KLA-Tencor Corporation. 160 Rio Robles, San Jose, California 95134, United States.
[c]SYSTEMA Systementwicklung Dipl.-Inf. Manfred Austen GmbH, Manfred-von-Ardenne-Ring 6, Dresden, Germany

ABSTRACT

Especially for advanced masks the reticle inspection operation is a very significant cost factor, since it is a time consuming process and inspection tools are becoming disproportionately expensive. Analyzing and categorizing historical equipment utilization times of the reticle inspection tools however showed a significant amount of time which can be classified as non productive. In order to reduce the inspection costs the equipment utilization needed to be improved. The main contributors to non productive time were analyzed and several use cases identified, where automation utilizing a SECS[1] equipment interface was expected to help to reduce these non productive times.

The paper demonstrates how real time access to equipment utilization data can be applied to better control manufacturing resources. Scenarios are presented where remote monitoring and control of the inspection equipment can be used to avoid setup errors or save inspection time by faster response to problem situations. Additionally a solution to the second important need, the maximization of tool utilization in cases where not all of the intended functions are available, is explained. Both the models and the software implementation are briefly explained. For automation of the so called inspection strategy a new approach which allows separation of the business rules from the automation infrastructure was chosen.

Initial results of inspection equipment performance data tracked through the SECS interface are shown. Furthermore a system integration overview is presented and examples of how the inspection strategy rules are implemented and managed are given.

Keywords: Factory Automation, Equipment Productivity, Equipment Performance Tracking, Remote Control, Inspection Strategy

1. INTRODUCTION

When looking at the mask manufacturing process especially for advanced masks, the inspection costs meanwhile make up the largest portion of the mask costs and are expected to further increase with higher resolution nodes. This is caused by both increase of inspection equipment cost and actual inspection time needed. For that reason it made sense to investigate if overall inspection costs can be reduced by improving equipment utilization and reducing inspection time through automation. Thus the primary motivation for considering the usage of the available SECS interface for the TeraScan systems was the goal to reduce inspection time and therefore reduce overall costs. The second motivation for

[1] SECS SEMI Equipment Communications Standard

applying the automation interface was the requirement to get real time information about the current equipment usage which can be further processed by other systems and not just historical reports.

2. BACKGROUND

The starting point for any improvement actions was the analysis to break down the equipment utilization time.

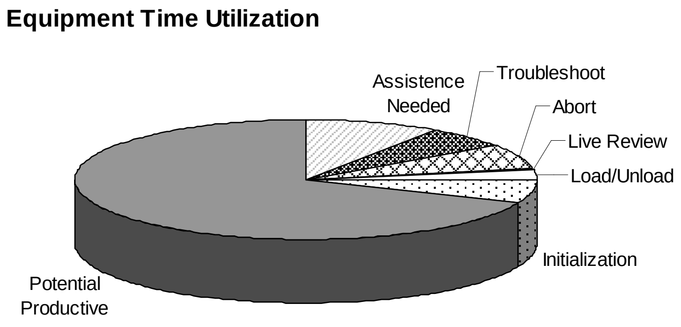


Figure 1: Typical average equipment time breakdown of a TeraScan system

The equipment utilization data shown in Figure 1 was generated using existing proprietary software which parses the inspection tool logs on a daily basis. The results of that analysis showed a significant amount of non productive time and its main categories. After that the reasons for the different non productive time categories and possible improvement actions were investigated. In order to identify appropriate actions the operational scenario of the inspection tool needed to be analyzed in detail.

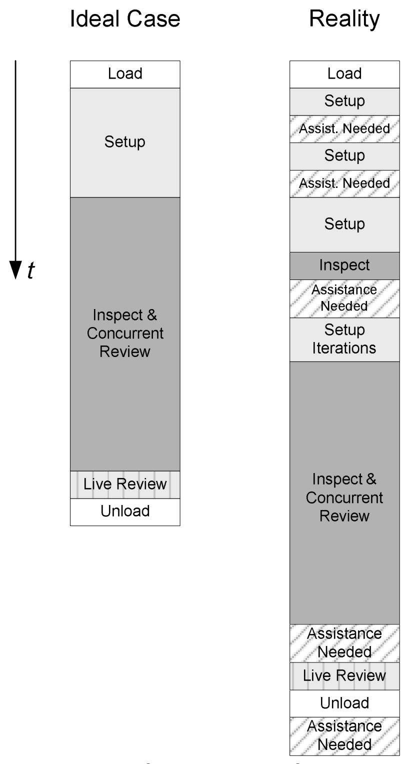


Figure 2: Simplified sequence of ideal versus common inspection run

In the ideal case shown on the left side of Figure 2 the mask needs some time to load, followed by several setup steps and then the actual inspection. Ideally the defects can be classified (in parallel) during inspection, what is called here concurrent review. If the defects cannot be classified based on the pictures taken, an additional live review part may follow. The sequence finishes with the mask unload.

However reality looks slightly different. During the setup process there are usually several steps where the operator has to enter data and then may need to wait for the tool to perform e.g. some alignment or calibration step. Since one operator needs to take care of multiple tools, time is lost until he is back to proceed with the next step. This causes the first Assistance Needed time listed in Table 1, while the actual data input is listed as Data Entry time.

An additional Assistance Needed situation can happen after the actual inspection or when the mask was unloaded. For identifying appropriate solutions it was assumed that Assistance Needed time can be reduced if an operator is immediately available when needed. This of course requires to have enough operators available and to have a way to immediately notify one when the tool needs help. For this to work the tool must be able to send appropriate notifications through its interface to the host.

The largest potential savings are in the Abort time category. This is basically wasted inspection time in case iterations for different reasons were needed. If the iteration is caused by incorrect data entries this could be avoided if the tool would accept remote setup and start. However this functionality was not yet available for the TeraScan system. That is why another option, to validate the setup parameters against the specification and thus to prevent starting with wrong setups, was pursued.

Non Productive Time Category	***Proposed Action***
Assistance Needed Time after alignment/ calibration step during setup time	notify operator when the step completed
Data Entry Time – The manual process of entering job data	remote setup and start from the host
Abort Time – Time wasted by jobs aborted because the set up was wrong	remote setup and start from the host or remote validation of setup
Assistance Needed Time after abort – the tool sits idle waiting for an operator after a job aborts	notify operator when the tool aborted a job
Assistance Needed Time after inspect – the tool sits idle waiting for the operator between inspection completed and defect review started	use concurrent review, otherwise notify operator when tool nears completion
Tool Empty Time – the tool is idle while an operator moves the last mask to the next operation and then gets the next mask to run	notify operator to fetch the next plate because an inspection is nearing completion

Table 1: Identified time categories which have potential for recapturing lost tool time and corresponding automation actions

3. SOLUTION

As the result of the business analysis two key use cases were identified, which where judged to have significant potential for improvement. The first case was to avoid or reduce operator Assistance Needed time. The second case was the maximization of tool utilization when the equipment has a problem. Below is a high level view of both scenarios. Thereafter these scenarios are discussed in more detail using the example of the inspection tool.

3.1 Reduction of Assistance Needed time

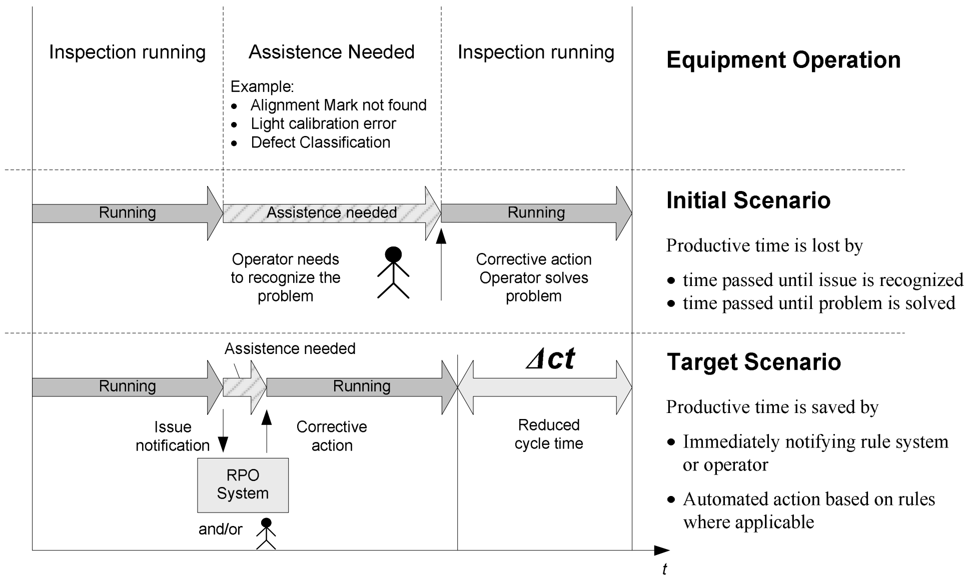


Figure 3: Approach for reducing cycle time and non productive time in the Assistance Needed scenario.

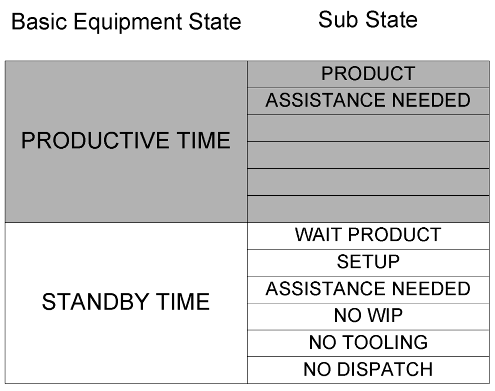
The upper row in Figure 3 shows the original tool usage scenario. First the inspection operation runs, then the tool stops for a particular reason and requires operator interactions. The sum of both, the time passed until the operator is aware of the problem and the time that is needed to get the tool back to operation, is counted as Assistance Needed Time.

In the second scenario shown in the lower part of Figure 1 the tool immediately sends a notification through its interface to the host system when it needs assistance. Now the notified operator or ideally in selected cases an automated system can respond. The reduction of this Assistance Needed time finally results in the desired decrease of cycle time.

3.2 Enhanced State Model

To reduce the Assistance Needed time it is desired to notify the operator immediately when a tool needs help. This requires to have that information available from the tool in real time and to have a system that is capable of processing that information. The latter capability required extending the original RAM[2] state model, which knew only the six basic equipment states. More over these states were only tracked by manual operator inputs and consequently that system was not feasible to track the need for an operator itself. The newly developed system supports automated tracking and supplemental sub states and thus allows a much more detailed tracking which is needed to decide the appropriate actions.

Figure 4 shows for example which sub states were introduced for the basic states Productive and Standby. Both basic states contain an Assistance Needed sub state, which can be used to trigger operator notification. It is obvious that this level of detail is only manageable when the state change information is automatically provided. Consequently the tools that are required to be monitored at this detailed level need to have a SECS interface. In case of the TeraScan inspection system this interface is available and provides sufficient details. With TeraScan SW R11.1.8 the host is capable of precisely tracking the processing states and material movements of the equipment. Based on this information the host can trigger the state changes between the different productive and standby states. By having that system in place it is now possible to get a detailed real time view of the tool utilization across all tools. Additionally this allows automatically initiating actions like notifying operators through automatically refreshed task list.



Basic Equipment State	Sub State
PRODUCTIVE TIME	PRODUCT
	ASSISTANCE NEEDED
STANDBY TIME	WAIT PRODUCT
	SETUP
	ASSISTANCE NEEDED
	NO WIP
	NO TOOLING
	NO DISPATCH

Figure 4: Example of additional RAM subs-states introduced

3.3 Preventing failed inspections

While the focus so far has been placed on minimizing Assistance Needed time, the next paragraph explains the solution to reduce Abort and Trouble Shooting time.

In normal operating mode of the TeraScan 5XX inspection tools several operator interactions are required. During common setups depending on mask and inspection type the operator may need to enter various parameters, and sometimes specifies up to 30 values for the detectors. This can be an error prone process. Ideally all of the data could be sent automatically through the equipment interface. Unfortunately the current interface SW of that tool does not support this yet, but it does allow to validate the setup data. This method can be used to ensure that only correctly set up inspections are processed.

[2] RAM Equipment Reliability, Availability, and Maintainability as defined in SEMI E10 [1]

Figure 5 displays the sequence diagram of the equipment and host interactions during the remote validation scenario. The left sequence demonstrates the correct parameter setup. After the host detects the completion of the setup, it compares the received setup data against the specification. If everything is OK it automatically starts the inspection run.

The right side diagram however shows the example case of an operator error. If host validation detects a mismatch, the host then sends an abort to the tool. Thus inspections with incorrect parameters due to typing errors can be prevented, which consequently saves Abort and Trouble Shooting time.

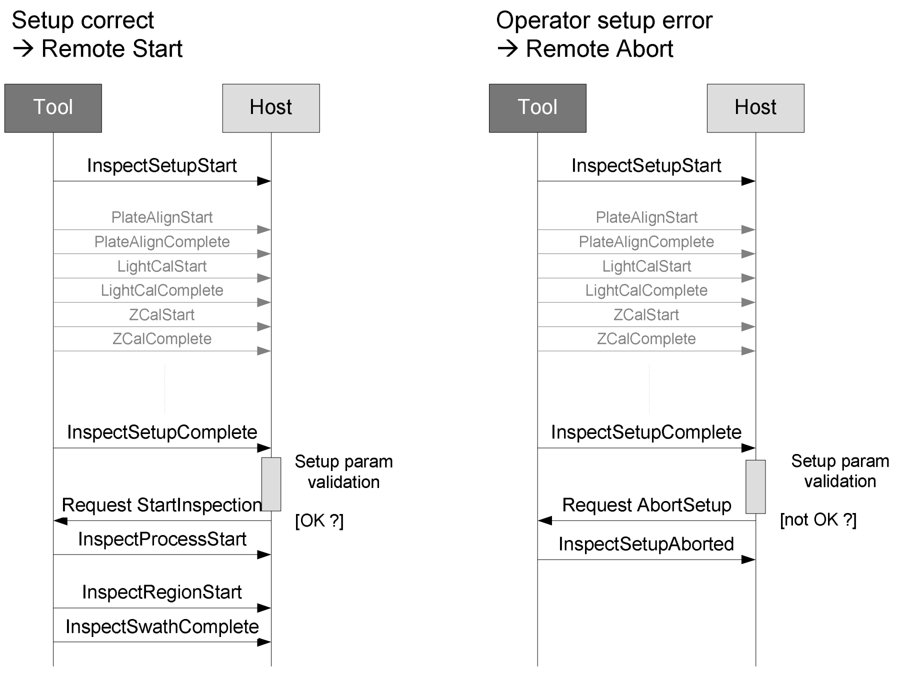


Figure 5: Remote validation scenario to avoid Abort and Trouble Shooting time

3.4 Maximization of tool utilization

The second important scenario for improving tool utilization is to individually handle situations such as where the equipment encounters a problem, but is still capable of performing its intended functionality for selected products. In this situation the tool is considered to have limited capability.

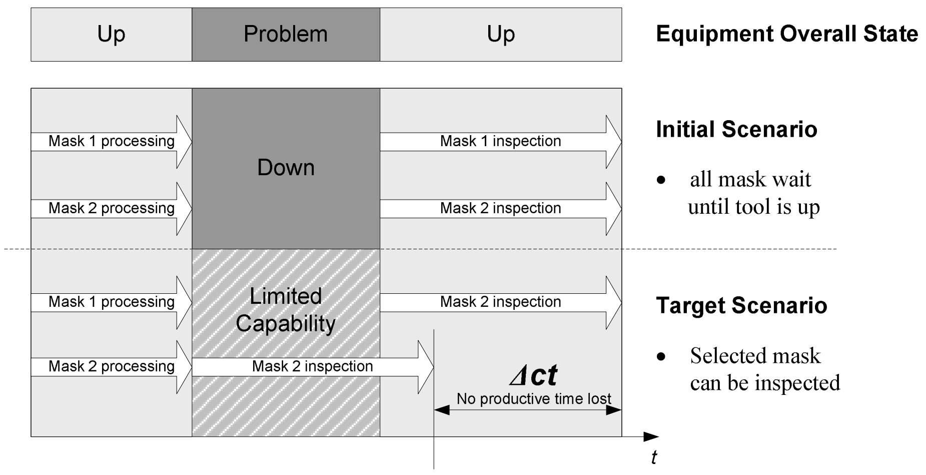


Figure 6: Solution to maximize tool utilization in the case that the equipment has a problem

The upper row in Figure 6 demonstrates the common initial scenario. Since the tool does not provide all of its intended functionality it is considered as down according to SEMI E10 [1] standard. That means that it would not be available for any inspections even in cases where the problem is known not to be relevant for the current product.

Therefore it was desired to deviate in this case from the rigorous SEMI E10 state definitions and to build a system which allows handling this limited capability situation. By introducing an appropriate capability model the automation software can distinguish and decide based on the mask specifications whether the tool is still good for a given inspection or not. Since for selected products the tool can still be utilized, higher tool utilization and a lower average mask cycle time is achieved in this second scenario.

3.5 Capability Model Concept

In order to model the limited capability situation a detailed state tracking of the available tool functions is needed. In case these functions do not directly depend on a physical equipment module like on a chamber etc. they are modeled as capability.

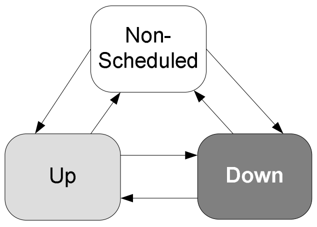


Figure 7 Capability State Model

For each tool, the tool owners decide which of the major equipment functionalities need to be individually tracked and then define an appropriate capability model.

Although these capabilities do not have a physical equivalent, they are considered as a virtual equipment module and thus can now be used to track the availability of relevant tool functions. Unlike the real equipment modules where six main states are tracked, the simpler three-state state model shown in Figure 7 was considered to be sufficient for capabilities.

These capability states will be tracked in addition to the common equipment module states which strictly follow the SEMI E10 [1] guidelines. Thus it is possible to provide the standard RAM[3] metrics, and additionally gain more detailed information for a more flexible production control.

Mode	Pixel			
	P125	P125R	P90	P90R
D2D	Up	Up	Up	Up
D2Db	Up	N/A	Down	N/A

Table 2: Example of the modeled capabilities of an inspection tool

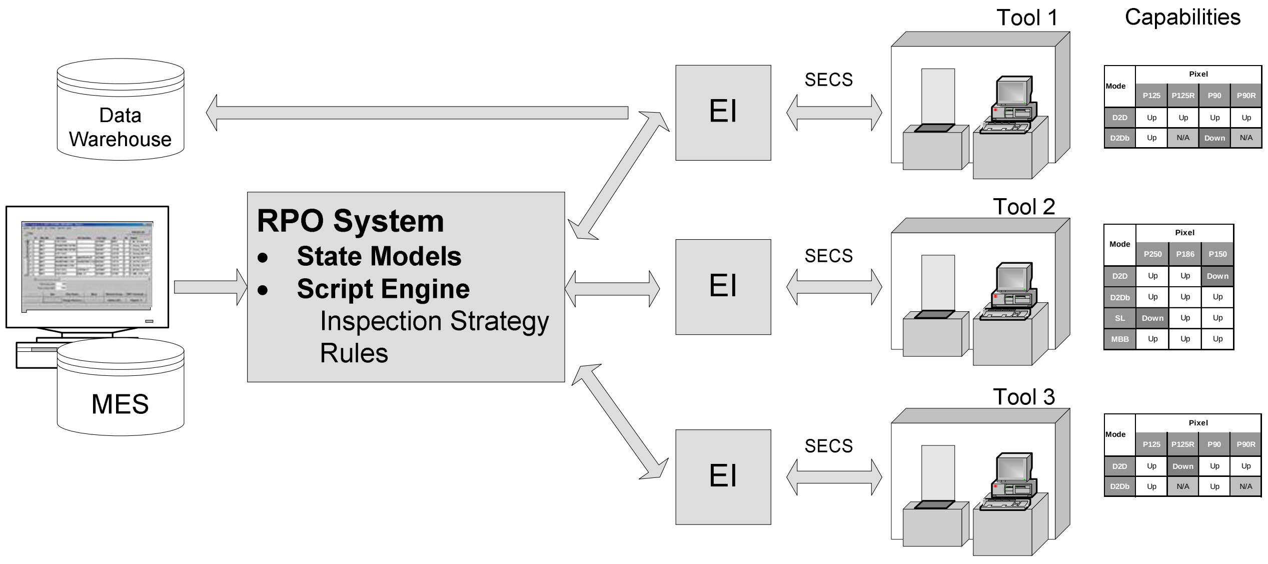
Table 2 shows a capability model example where the different inspection algorithms and pixel sizes of an inspection tool are modeled and thus can be individually tracked and evaluated.

[3] RAM Equipment Reliability, Availability, and Maintainability defined in SEMI E10 [1]

4. SYSTEM INTEGRATION

4.1 Automation software implementation

Figure 8 provides a visual diagram of how the capability model is actually used to maximize tool utilization. The central component is the RPO[4] system which implements both the equipment and the capability state tracking, and also provides the inspection strategy application. The latter contains a set of rules and tables and decides based on both, the mask specification parameters and the states of the capability and equipment model, where the mask should be inspected and which setup parameters are to be used. It can also decide to split up inspections across several tools.



Mode	Pixel			
	P125	P125R	P90	P90R
D2D	Up	Up	Up	Up
D2Db	Up	N/A	Down	N/A

Mode	Pixel		
	P250	P186	P150
D2D	Up	Up	Down
D2Db	Up	Up	Up
SL	Down	Up	Up
MBB	Up	Up	Up

Mode	Pixel			
	P125	P125R	P90	P90R
D2D	Up	Down	Up	Up
D2Db	Up	N/A	Up	N/A

Figure 8: High level overview of how RPO interacts with MES, EI[5] and equipment

The information round trip scenario works in the following way. The operator initiates the inspection from the MES[6] system, and then the RPO rules will select the tool and determine the parameters to be used. Next the operator enters all the data at the selected equipment and performs the required setup steps. At the end of the setup the host validates the entered parameters and, if setup parameter verification succeeds, starts the actual inspection.

During the whole process the equipment interface software keeps track of all the tool activities and forwards the relevant information to the RPO state model. Finally when the host notices the end of the inspection it triggers the upload process of the inspection report to the data warehouse.

4.2 Business Rule Management via Domain Specific Language and Domain Model

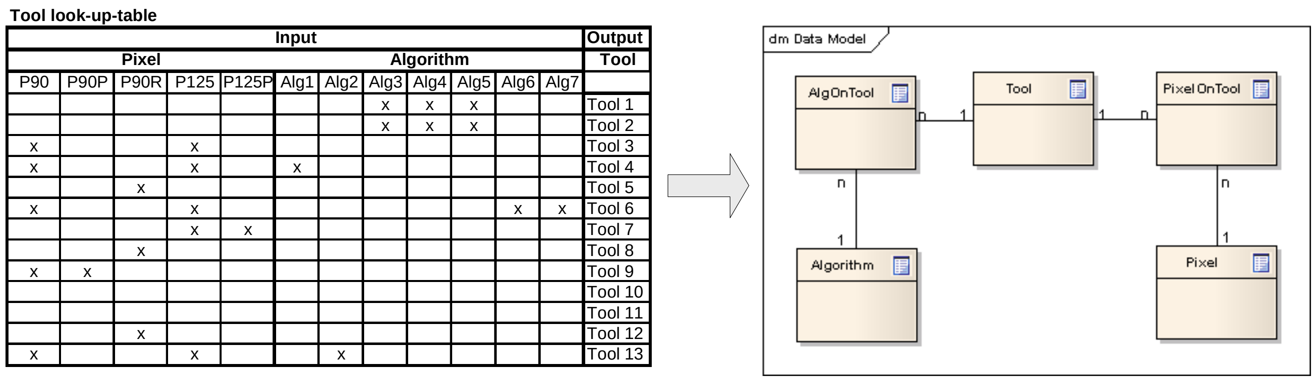
The concept of the inspection strategy was previously implemented at the AMTC, but had been done as stand alone application without automation interfaces. In order to be able to automate the information flow the original application needed to be migrated to a system which provides appropriate interfaces. More over it was desired to decouple responsibility and change cycles for the software infrastructure from that of the business rules processed by the inspection strategy.

One of the common approaches to meet such requirements is the use of a business rules management system (BRMS). After evaluating different available implementations it was judged that the effort of integrating such system which complies with the OMG[7] specification [3] would be inadequately high and would not even cover the complete existing

[4] RPO Rule Based Productivity Optimization System
[5] EI Equipment Interface - proprietary software which automates information exchange between equipment and other manufacturing software systems
[6] MES Manufacturing Execution System
[7] OMG Object Management Group

inspection strategy logic. For that reason the concept of a Domain Specific Language (DSL) in combination with a Domain Model was chosen. This allows creating a small special purpose language which is expected to be simple enough to be usable and maintainable by the business owners - in this case the inspection engineers.

The developed solution provides software tools to graphically build and change the domain model (the look up tables) without changing the underlying database structure by using a Meta-Model description (see Figure 9). This means that the database does not actually store the Domain tables; it stores how the tables look like.



Tool look-up-table

Input												Output
Pixel					Algorithm							Tool
P90	P90P	P90R	P125	P125P	Alg1	Alg2	Alg3	Alg4	Alg5	Alg6	Alg7	
							x	x	x			Tool 1
							x	x	x			Tool 2
x			x									Tool 3
x			x		x							Tool 4
		x										Tool 5
x			x							x	x	Tool 6
			x	x								Tool 7
		x										Tool 8
x	x											Tool 9
												Tool 10
												Tool 11
		x										Tool 12
x			x			x						Tool 13

Figure 9: Example of migrating the original lookup table to normalized Meta-Model table structure

The desired separation of the business logic from the automation infrastructure was implemented with a script engine which can manage and run business rules written in this DSL. Figure 10 shows a high level overview of the components developed to implement this script engine. In order to simplify the business rule creation and management as much as possible a limited and problem specific set of language key words was developed as part of the Domain Specific Language definition. The DSL definition itself was written using the Groovy scripting language similar as described in [4] and the resulting rules run on a Java EE Middleware platform. It is expected that after the initial migration of the inspection strategy logic these DSL based business rules can be managed by the responsible business owners.

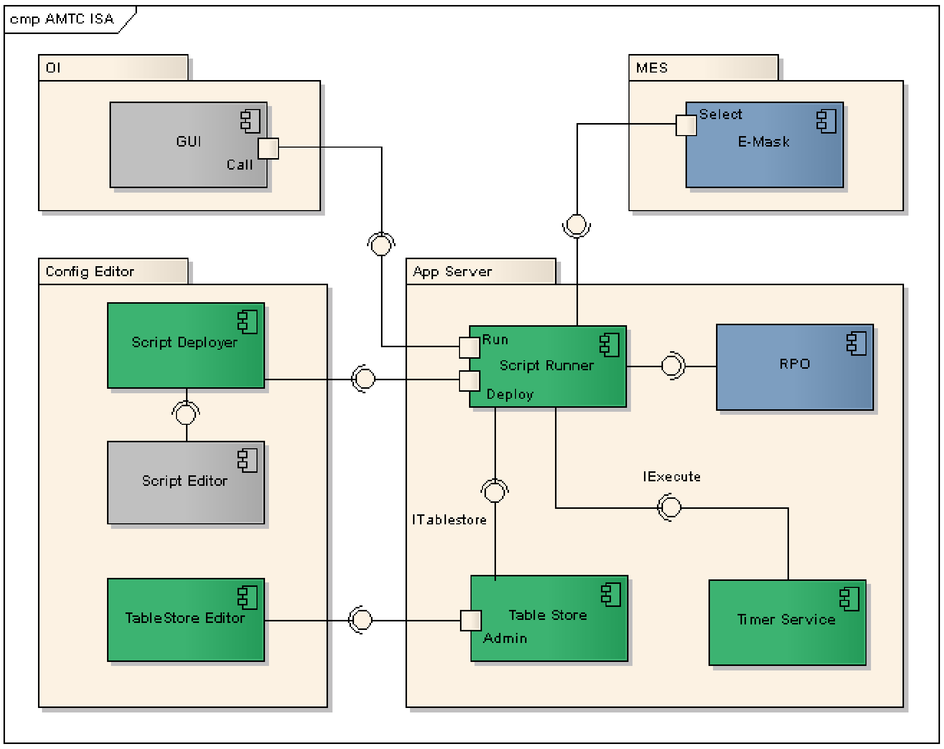


Figure 10: Component diagram of script engine solution for managing Business Rules

5. RESULTS AND OUTLOOK

The new developed automation solution is currently in the state of integration testing and most of its services run in parallel to the established system, which solely relied on operator inputs. That means that it collects data and tracks information but does not actively influence the production process yet.

All of the TeraScan 5XX systems at the AMTC were integrated via the SECS interface and reliably deliver detailed status information. It was shown that equipment utilization data can be precisely measured in real time (see also Table 3). Additional work is going on to automate the information flow for the state tracking of the remaining basic RAM states to provide the desired sub state level details. This required the development of new interfaces to the software systems used to manage maintenance (scheduled down sub states) and repair (unscheduled down sub states) situations. One of the next steps will be to activate the remote setup validation and the remote start once the new inspection strategy application is productive.

TIMESTAMP	TOOL_EVENT	RPO_EVENT	STATENAME	DURATION
10.12.2008 04:43:07	LibPlateToStageLoadStart	BEGIN_RUN	UP.PRODUCTIVE.PRODUCT	0:02:38
10.12.2008 04:45:45	LibPlateToStageLoadComplete	PAUSE_RUN	UP.PRODUCTIVE.NEED_ASSISTENCE	0:00:27
10.12.2008 04:46:12	PlateAlignStart	CONTINUE_RUN	UP.PRODUCTIVE.PRODUCT	0:02:29
10.12.2008 04:48:41	PlateAlignComplete	PAUSE_RUN	UP.PRODUCTIVE.NEED_ASSISTENCE	0:01:06
10.12.2008 04:49:47	LightCalStart	CONTINUE_RUN	UP.PRODUCTIVE.PRODUCT	0:01:10
10.12.2008 04:50:57	LightCalComplete	PAUSE_RUN	UP.PRODUCTIVE.NEED_ASSISTENCE	0:00:21
10.12.2008 04:51:18	ZCalStart	CONTINUE_RUN	UP.PRODUCTIVE.PRODUCT	0:01:06
10.12.2008 04:52:24	ZCalComplete	PAUSE_RUN	UP.PRODUCTIVE.NEED_ASSISTENCE	0:00:15
10.12.2008 04:52:39	InspectingRegionStart	CONTINUE_RUN	UP.PRODUCTIVE.PRODUCT	0:06:45
10.12.2008 04:59:24	InspectingRegionComplete	PAUSE_RUN	UP.PRODUCTIVE.NEED_ASSISTENCE	0:01:48
10.12.2008 05:01:12	DefectReviewStart	CONTINUE_RUN	UP.PRODUCTIVE.PRODUCT	0:32:26
10.12.2008 05:33:38	DefectReviewComplete	PAUSE_RUN	UP.PRODUCTIVE.NEED_ASSISTENCE	0:03:59
10.12.2008 05:37:37	InspectSetupStart	CONTINUE_RUN	UP.PRODUCTIVE.PRODUCT	0:02:38
10.12.2008 05:40:15	PlateAlignComplete	PAUSE_RUN	UP.PRODUCTIVE.NEED_ASSISTENCE	0:00:12
10.12.2008 05:40:27	LightCalStart	CONTINUE_RUN	UP.PRODUCTIVE.PRODUCT	0:01:15
10.12.2008 05:41:42	LightCalComplete	PAUSE_RUN	UP.PRODUCTIVE.NEED_ASSISTENCE	0:00:06
10.12.2008 05:41:48	ZCalStart	CONTINUE_RUN	UP.PRODUCTIVE.PRODUCT	0:01:03
10.12.2008 05:42:51	ZCalComplete	PAUSE_RUN	UP.PRODUCTIVE.NEED_ASSISTENCE	0:00:13
10.12.2008 05:43:04	InspectingRegionStart	CONTINUE_RUN	UP.PRODUCTIVE.PRODUCT	0:05:55
10.12.2008 05:48:59	InspectingRegionComplete	PAUSE_RUN	UP.PRODUCTIVE.NEED_ASSISTENCE	0:00:49
10.12.2008 05:49:48	InspectSetupStart	CONTINUE_RUN	UP.PRODUCTIVE.PRODUCT	0:02:38
10.12.2008 05:52:26	PlateAlignComplete	PAUSE_RUN	UP.PRODUCTIVE.NEED_ASSISTENCE	0:00:12
10.12.2008 05:52:38	LightCalStart	CONTINUE_RUN	UP.PRODUCTIVE.PRODUCT	0:01:13
10.12.2008 05:53:51	LightCalComplete	PAUSE_RUN	UP.PRODUCTIVE.NEED_ASSISTENCE	0:00:13
10.12.2008 05:54:04	ZCalStart	CONTINUE_RUN	UP.PRODUCTIVE.PRODUCT	0:08:15
10.12.2008 06:02:19	ZCalComplete	PAUSE_RUN	UP.PRODUCTIVE.NEED_ASSISTENCE	0:00:19
10.12.2008 06:02:38	InspectingRegionStart	CONTINUE_RUN	UP.PRODUCTIVE.PRODUCT	2:16:24
10.12.2008 08:19:02	DefectReviewComplete	PAUSE_RUN	UP.PRODUCTIVE.NEED_ASSISTENCE	0:01:43
10.12.2008 08:20:45	StagePlateToRSPUnloadStart	CONTINUE_RUN	UP.PRODUCTIVE.PRODUCT	0:05:27
10.12.2008 08:26:12	PlateToRSPUnloadComplete	END_RUN	UP.STANDBY.NEED_ASSISTENCE	0:06:35

Table 3: Example of an inspection run automatically tracked via equipment interface

Although the collected data are not yet used for the actual production control, they already helped to identify weak spots and improvement activities. In the next step both the user interfaces and the OEE[8] reporting needs to be completely migrated to use the new system, before a significant business impact can be expected. Based on the experiences collected while running the prototype system, we expect a significant reduction of this operator Assistance Needed time once the operator notification becomes effective.

[8] OEE Overall Equipment Efficiency as defined in SEMI E79 [2]

Looking at the inspection tool operation itself there is further potential seen for saving setup time by automating this step through the SECS interface once the equipment software supports this. However the highest benefit out of the automation could be achieved if the need for manual interactions during the setup process could be avoided at all. This means that the recipe and template creation process for these inspection tools requires enhancements to allow the automation software to provide all required inputs for the calibration and alignments based on chip design data upfront.

6. CONCLUSION AND SUMMARY

This paper has presented an automation solution which addresses the need to limit the increasing inspection costs by improving equipment utilization. The work explains the identified improvement actions and details how an integrated software solution can support there implementation. The paper illustrates the developed concepts and how they can be implemented using software technology. Further automation steps envisioned and current limitations e.g. from equipment side are discussed too.

The initial results utilizing the new system were presented and look promising such that a significant business benefit out of the automation efforts is expected.

7. ACKNOWLEDGEMENTS

AMTC is a joint venture of AMD, Qimonda and Toppan Photomasks and gratefully acknowledges the financial support by the Free State of Saxony in the framework of the technology grants based upon European Regional Development Funds and funds of the Free State of Saxony under contract number 12251/1976.

REFERENCES

1. SEMI E10-0304E Specification for Definition and Measurement of Equipment Reliability, Availability, and Maintainability (RAM), SEMI International Standards, 3081 Zanker Rd., San Jose, California 95134, USA, 2007
2. SEMI E79-1106 Specification for Definition and Measurement of Equipment Productivity, SEMI International Standards, 3081 Zanker Rd., San Jose, California 95134, USA, 2007
3. Semantics of Business Vocabulary and Business Rules (SBVR), v1.0 Object Management Group, Inc. (OMG), http://www.omg.org/docs/formal/08-01-02.pdf
4. Writing Domain-Specific Languages, http://docs.codehaus.org/display/GROOVY/Writing+Domain-Specific+Languages