Application Development Framework for Measurement-based Product Testing Management

Zhongwen Guo, Yu Yu, Zhaosui Sun and Yunhong Lu
Department of Computer Science and Engineer, Ocean University of China, Qingdao 266100, China
Email: {guozhw2007, rainertop}@gmail.com, {i2mtc_zhaosui, i2mtc_yunhong}@163.com

***Abstract*— Measurement-based product testing management systems (MPTMS) are widely used during the life cycle of products to validate the design and test the products. Unfortunately, measurement data from measurement systems are isolated with testing task information in the MPTMSs. Furthermore, the development of MPTMSs often begins from scratch, which leads to low development efficiency and low reusability. To solve the problems mentioned above, an application development framework is designed. The framework supports seamless integration between MPTMSs and measurement systems. Meanwhile, components like resource management, organization management, workflow management, testing task management and process mining are provided to improve the reusability of the framework. The development process of MPTMSs consists of workflow model design, components reuse and user interfaces customization. Moreover, an application example is given to validate the framework and the result shows its availability.**

***Index Terms*— Measurement-based product testing management systems, measurement systems, application development framework, workflow management, process mining**

I. Introduction

MPTMSs are becoming pervasive in factories because of quality management. The tests in MPTMSs are performed on measurement systems with product samples. By managing the measured data, control instruments and sensors, measurement systems composed of sensors, instrument and computers can perform measurement operations. The measured data from the measurement system are basis for test results in MPTMSs. MPTMSs manage product's quality in life-cycle by enforcing tests against test standards which specify test items with test result judgment rules.

As for product testing, challenges are listed as follows:

- Data sharing: Measurement data are from heterogeneous measurement systems. It is very hard to incorporate the measurement data into MPTMSs. However, shortages of corresponding measurement data make test results lack of credibility. Meanwhile, measurement systems are isolated from testing task information, which makes measurement systems separate from MPTMSs.
- Workflow management: In product manufacturing factory, many kinds of product's testing management systems coexist. However, those systems are lack of extendibility, which causes redundant construction and low development efficiency.
- Testing task management: Devices, test engineers and product samples are dynamic changing resources, which increases the complexity of testing task management for test order and test report.
- Process mining: MPTMSs generate event logs for every running workflow instances to trace the processes. Those logs should be mined to obtain valuable information for process improvement or problem identification.

Most of previous work focuses either on development of measurement systems [1]–[10] or on human labor cost and resource management involved [11]–[14]. Aimed at large system-level test and diagnostic related systems, IEEE SCC20 community develop a serial of standards which include hardware interfaces(HI), diagnostic and maintenance control(DMC), test and ATS description (TAD), test information integration (TII). However, it cannot be used for product level test described in this paper, which puts more emphasis on measurement data and testing task information. Testing management system (TingMS) of YOKOGAWA provides functionalities of test management, data acquisition, curve browse, report export and instrument management. However, it cannot provide management to tests from a workflow-centric perspective. There are few studies on testing management systems, especially in MPTMSs area. It is difficult for existing solutions to solve the challenges of measurement-based product testing management. It is necessary to design and implement an application development framework for MPTMSs. In this paper, the requirements of MPTMSs are analyzed and an application development framework is proposed first. Then, based on such framework, design strategy and implement details are given. To construct a new MPTMS easily, we implement it by deploying a new workflow model [15], database tables, user interfaces along with reusable components of the framework. Specially, our main contributions are as follows:

This paper is based on "An Integrated Application Framework for Measurement-Based Product Testing Management," by Yu Yu, Zhongwen Guo, Zhaosui Sun and Yunhong Lu, which appeared in the Instrument and Measurement Technology Conference, 2012 IEEE International, Graz, Austria, May 2012. © 2012 IEEE.

doi:10.4304/jsw.8.6.1374-1383

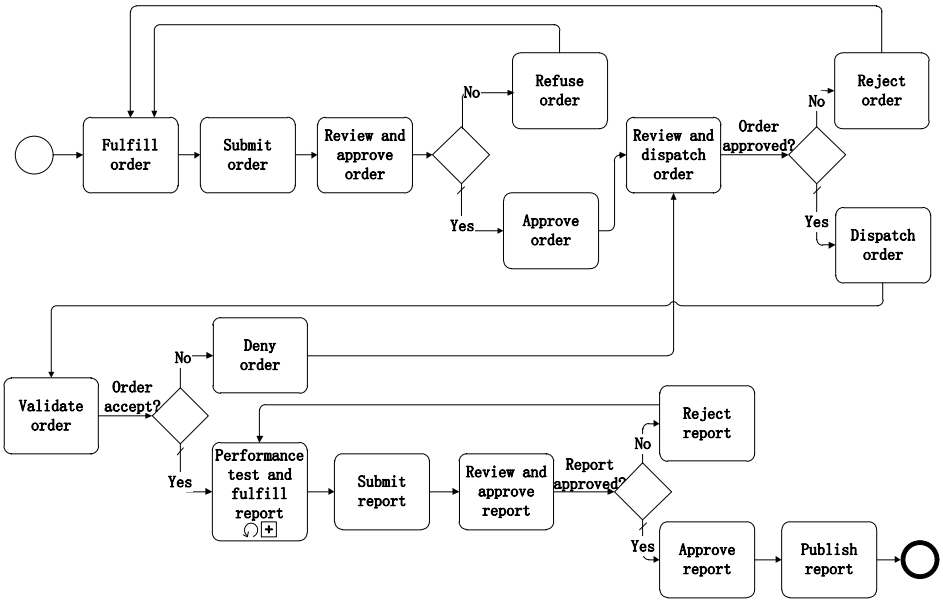

Figure 1. A generic measurement-based product testing process

- The process requirements for measurement-based product testing management are studied and further decomposed into roles, actions and data entities.
- An application development framework for measurement-based product testing management is proposed, which support agile system development by leveraging on scalable architecture and reusable components.
- Process mining is used to further discover secret behind system event logs, which supports continue process improvement.

The reminder of the paper is structured as follows. Section II outlines background of MPTMSs. Section III describes the application development framework for MPTMSs. Design of the framework is presented in Section IV. An application example is given in Section V. Finally, conclusions are given in Section VI.

II. Background

Fig.1 demonstrates the generic process of measurement-based product testing using BPMN notation [16]. The main participants in measurement-based product testing are R&D engineer, R&D project manager, lab supervisor, test engineer and test supervisor. At first, R&D engineer fills test order and submits the test order. The test order contains test items to be tested against standards. Then, the order will be reviewed by R&D project manager. If the order is approved by the R&D project manager, the order will be sent to lab supervisor. In response to the test order, lab supervisor checks the test order. If the check is passed, the test order will be received by lab supervisor. Then lab supervisor dispatches the test order to a certain test engineer. Then test is performed and measurement data are collected by measurement system. Test report is filled by test engineer and submitted to test supervisor. Upon receiving the test report, test supervisor checks it by his/her experience and examining measurement data. The test report will be published if it passes the check. Then, R&D engineer can view test results in the test report.

This work is based on our previous work on measurement systems. In [17], a software generation platform for DMS [18] development in the household appliance test field has been developed. A measurement system architecture which offers a universal client with rich user interaction, facilitates system integration and simplifies the system development has been described in [19]. Given continuity to those works, we further design an application development framework in this paper, which supports agile software development of MPTMSs and integration with measurement systems.

III. Measurement-based product testing management application development framework

In the measurement-based product testing management area, there is no existing application development framework that guides the development of MPTMSs. MPTMSs

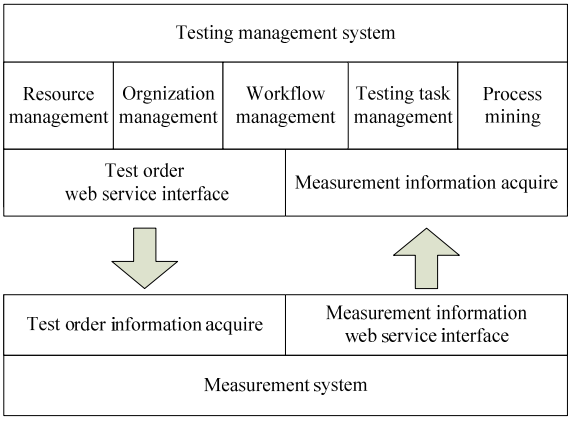

Figure 2. MPTMS application development framework

from various manufacturers are developed on special solutions, which leads to low development efficiency. In this section, a MPTMS application development framework is provided (See Fig. 2).

Most of the MPTMSs follow the same structure: users login in the system via user portal. Based on responsibilities, users are classified into different roles. Multiple roles are involved to accomplish a test order. Information passed through those roles is controlled by workflow management component. Besides that, there are some components which are essential for product testing management, such as resource management, organization management and testing task management. The scalability of architecture is achieved by taking cloud computing [20] into consideration from the very beginning of system design. By leveraging on cloud computing utilities, system can be easily horizontally extended by using multiple virtual machine instances. Loading balance ability can be provided by databases clustering and app servers clustering in those virtual machine instances. Besides that, the agility of the framework lies in its abilities to handle rapid user demand fluctuation. Advanced web application develop technology is used to promote the reusability of those components, such as ExtJS framework [21]. Building and modifying of user interfaces are directly through WYSIWYG [22] web builder. Another key point of MPTMSs is integration with measurement systems. Detailed description about integration is in [23].

A. MPTMS

- *Resource management*: Resource management component maintains resources shared by product testing workflow instances, such as product sample information. Product sample information is used to describe product sample characteristics. It should be fulfilled first and then included in each test order. In order to improve efficiency and reduce mistakes in product sample information fulfilling, most of information is maintained previously in the system by administrator, such as product sample type, level, technical parameters and main parts. Most of user operations in product sample information fulfilling are searching and selecting items from existing information list.
- *Organization management*: An organization is composed of users, departments, roles and authorities. Concretely, users belong to zero or more roles and departments. Role-based administration to authorities is used to protect the security of important and sensitive data. Organization management provides the function of modeling organization and organization information query. Organization modeling is often accomplished by administrator. Then, the function of organization information query is invoked by workflow management component to acquire organization information, such as which department a user belongs to and the supervisor of this department.
- *Workflow management*: Workflow management component orchestrates functions of other components. Tests are abstracted as user reaction driven running workflow instances which assembles all the tests needed information. However, user can only drive the workflow instances in a predefined way following the process model. Process model contains activities and connections between those activities. Those activities are with constrains of user, role and organization. Connections are with condition for route selection. Workflow provides maximum flexibility for accommodating to ever-changing product testing requirements while requires minimum modifications to existing systems.
- *Testing task management*: Testing task management component provides management to test order and test report. User's routine operations are heavily depended on testing task management. Testing task management delivers test orders and test reports to users with certain permitted operations according to the workflow instances related to those test orders and test reports. It also provides the function of test scheduling. Test scheduling improves the efficiency of product testing management which constrains by dynamic changing resources.
- *Measurement information acquire*: Measurement data are the basis of test results. Measurement information acquire component gets measurement data from measurement systems by invoking the measurement information web service interfaces provided by measurement systems.
- *Process mining*: The pipeline of process mining begins from system event logs. Those logs are transformed into MXML log format supported by ProM [24]. ProM is a process mining tool which extracts and presents the information typically needed in measurement-based product testing management. Event logs transformation is supported by process mining component. Besides that, it also provides information query functions to processes related data. To further understand process mining result, users have to turn to information query function.
- *Measurement systems*: Measurement systems are used by test engineers to perform the test according to standards information in the test order. In a common scenario, the product sample is attached with sensors. Then, product sample is running according to requirements of standards. Data collected by sensors are stored and further handled by measurement system. Usually, data are presented in an intuitive way, such as curves with colors and marks in a coordinate system. Curves can be included in test report to certify the test result.

B. Interfaces between MPTMSs and measurement systems

- *Web service interface of measurement information* : Web service interface of measurement information complied with IEEE 1851 standard which defines system framework, interface of web services, and

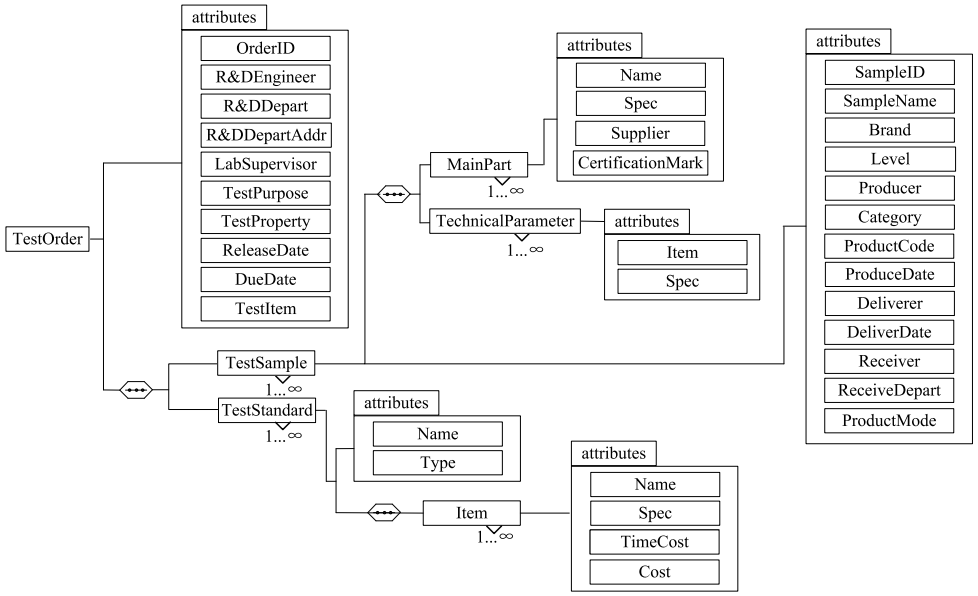

Figure 3. Test order information in the schema

data exchange formats that facilitates inter-operation and data sharing between measurement systems. IEEE 1851 standard was formally approved by IEEE at June 8, 2012. The main content of the standard are from our R&D team and scientific research. Detailed information about IEEE 1851 standard can be found in [25].

- *Web service interface of test order information* : In Fig.3, the schema for test order information is depicted. The order information schema is summarized from years of experience with user requirements and measurement-based product testing systems. Test order information composes of three parts: test order properties, test sample information and test standard information.
  - *Test order properties*: Test order properties describe the main attributes of test order. Detailed descriptions of those attributes are listed in Table I.
  - *Test sample information*: Test sample information describes under testing product. It composed of test sample properties, main parts and technical parameters. Detailed descriptions of those are given in Table II.
  - *Test standard information*: Test standard is unique identified by its name. It can be national standard or company internal standard. Each standard contains many items specify test names and specifications the tests should be followed.

IV. Design of Measurement-Based Product Testing Management Application Development Framework

A. Design strategy

Based on the analysis of user requirements, workflow management is adopted in the system design. System is partitioned into components which serve as information provider for workflow instances. A test begins from test order and ends at test report. During the process of test, users of different roles cooperate in a predefined way which prevents mistakes from happing. Basically, there are two important factors should be considered during the process of system design.

- *Statuses of test order and test report*: The status of a workflow instance changes when the status of its activities changes. From the view of users, test order and test report also have different statuses, such as submitted order, check pending order, audited order. The mapping between statuses of test order, test report and statuses of workflow instances will certainly reduce the complexity of system design.
- *Operations of different roles*: Users operations are associated to its roles. Roles can have different operations according to the activities they can participate in. So, mapping between activities in workflow instances and roles operations should also be maintained. Based on the mapping, system can be constructed with user interfaces classified by roles. System will automatically load different configuration to initialize different roles user interface, which guarantees the reusability and scalability of component.

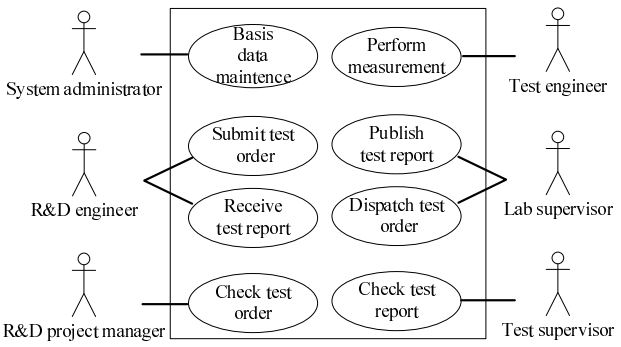

Figure 4. Use case diagram

TABLE I.
TEST ORDER PROPERTIES

Attribute	Description
OrderID	OrderID is the unique identification of test order
R&DEngineer	R&DEngineer is the originator of test order
TestPurpose	TestPurpose indicates purpose of test, such as product validation test, proof test, type test, product development test
TestProperty	TestProperty includes entrustment test and sampling test

TABLE II.
TEST SAMPLE INFORMATION

Attribute	Description
MainParts	MainParts are the main constituent parts of the test sample
TechnicalParameters	TechnicalParameters describe sample's technical characteristics, such as voltage rating, power rating and current rating
Level	Level specifies current level in product's whole lifecycle. It can be process prototype, manual prototype, small batch prototype and batch products
ProductMode	ProductMode is the unique identification to mode of the product. A product of certain mode may have test samples of different levels

B. Role definitions

In product testing management, workflow instances may involve multiple users which belong to different roles. In this subsection, detailed roles explanations are given (See Fig. 4).

- System administrator: System administrator is responsible for basis data maintenance. Basis data are fundamental to the product testing management. Information in test order and test sample, such as test purpose, test property, level, test standard information, category, productMode, productCode, main parts and technical parameters are maintained as basis data by the system administrator.
- R&D engineer: R&D engineer uses MPTMSs to submit test order and get test report.
- R&D project manager: R&D project manager is the leader of R&D engineers. Test orders submitted by R&D engineers will be checked by R&D project manager.
- Lab supervisor: The test orders checked by R&D project manager will be checked again by lab supervisor. If check is failed, the test order with problems specified is returned to R&D engineer who should solve the problems and submit the test order again. Then, the test orders are dispatched to test engineers by lab supervisor.
- Test engineer: Tests are carried out by test engineers according to test standard items specified in the test orders. Test engineers fulfill test report according to measurement data from measurement system. Then, the test reports are submitted to test supervisor for checking.
- Test supervisor: Test reports are checked by test supervisor. If test report passes the check, the test report will be published. Then, R&D engineer can get the published test report.

C. Organization information

Organizational management is vital to workflow applications [26]. Organization information schema is provided

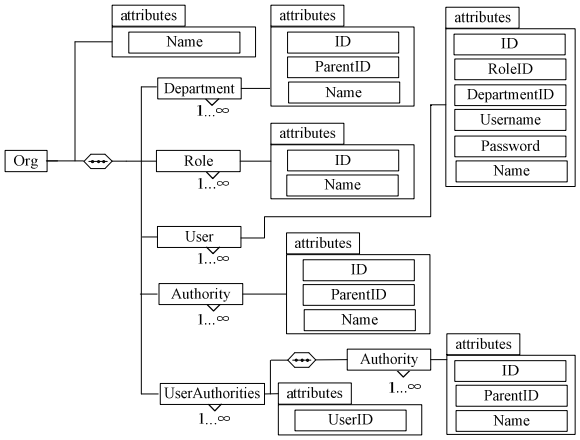

Figure 5. Organization information in the schema

in Fig. 5. The organization information schema includes department information, role information, user information, authority information and user authority information. The schema supports departments organized in nested form. Each department can have many sub departments. Users can belong to one or many departments and roles. All authority items in the system are contained in authority information. User authorities associate user with corresponding authority items.

D. Workflow management

In order to improve the reusability of workflow management component. A workflow engine is designed. Building a new workflfow engine other than leveraging on exist workflow engines like Enhydra Shark [27] and jBPM [28], because those workflow engines are hard to be customized. The workflow engine is composed of process definition and runtime environment. Detailed information about process definition and runtime environment will be described below.

- *Process definition* is translated into XML based definition file which complies with corresponding XML schema (see Fig. 6). Each process definition consists of one process element which has name attribute. The process element consists of eight types of sub

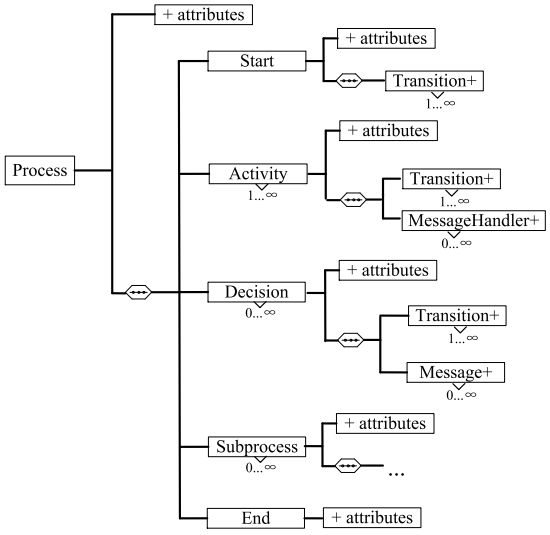

Figure 6. Process definition in the schema

elements which can be further classified into flow objects (Start, Activity, Decision, Subprocess and End), connections (Transition) and messages(Message and MessageHandler).

- Transition: Transition has attributes of name, from and to. It refers to the path from one flow object to another.
- Start: Start element contains at least one transition element which refers to entrance of the process.
- Activity: Activity element contains at least one transition element which refers to work items in the process.
- Decision: Decision element contains transition elements with condition attributes. It refers to a selection made by users, which decides the path in the process.
- Message: Messages are used to reduce the complexity of flow control. A message is sent in response to special condition in decision.
- MessageHandler: Message handler responses to messages by amending statuses of activities and routing flow in workflow instance.
- Subprocess: Subprocess is circular definition of process element. Multiple subprocesses elements can be included in a process.
- End: End element has name attribute which refers to the end of process.

- *Runtime environment* provides execution API (Application Programming Interface) for driving workflow execution. Architecture of runtime environment is depicted in Fig. 7. Process XML definition is checked by schema validation in schema validation sub component. Then, process XML definition is resolved into elements. Then those elements will be persistent into corresponding database tables by model persistence sub component. A new workflow instance is created by calling corresponding execution API exposed by execution component. The execution of workflow instance is driven by calling corresponding execution API.

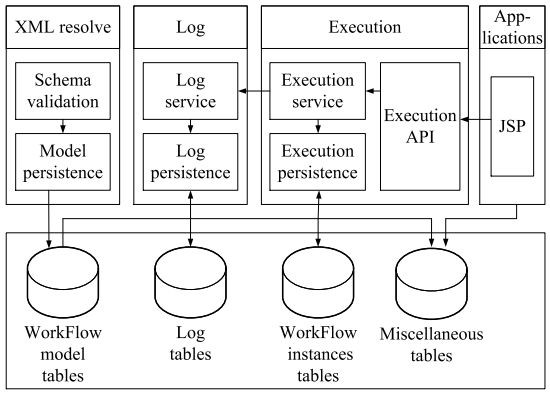

Figure 7. Runtime environment architecture

TABLE III.
TEST DATA FOR JOB SHOP SCHEDULING

Jobs	Machine sequence	Processing times
1	$1, 2, 3, 4$	$p_{11} = 12, p_{21} = 110$ $p_{31} = 70, p_{41} = 75$
2	$2, 1, 4, 3, 5$	$p_{22} = 110, p_{12} = 12$ $p_{42} = 75, p_{32} = 70, p_{52} = 120$
3	$1, 2, 4$	$p_{13} = 24, p_{23} = 180$ $p_{43} = 100$

E. Testing task management

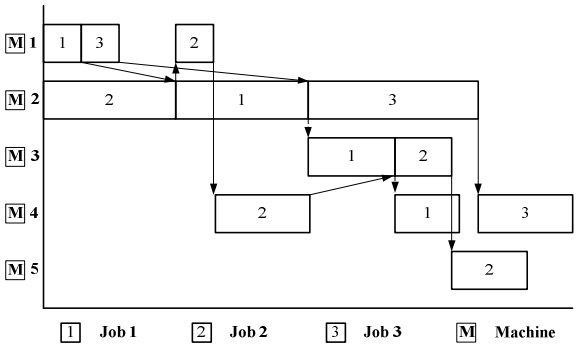

Figure 8. Gannt chart: a feasible solution to the test data

Machine 1	Start	Stop
Job 1	0	12
Job 3	12	36
Job 2	110	122
Machine 2		
Job 2	0	110
Job 1	110	220
Job 3	220	400

Machine 3	Start	Stop
Job 1	220	290
Job 2	290	360
Machine 4		
Job 2	122	197
Job 1	290	365
Job 3	400	500
Machine 5		
Job 2	360	480

Figure 9. Dispatch list interface of the solution

Testing task management provides functionalities of test order management and test report management. Test order management includes edit order, submit order,

check order, save order and view order. It includes all the operations to test order and dispatches them to user according to user role. In addition, test report management includes edit report, submit report, check report, and view report. It also includes all the operations performed on test report and dispatches them to the user according to user role.

Besides that, testing task management also supports test scheduling which will be emphasized on. The aim of test scheduling is to schedule all the tests so as to minimize the earliest time by which all the tests are completed. We relate the test scheduling problem with deterministic job shop scheduling problem [29]. Test orders can be viewed as jobs.

Each job consists of n operations, corresponding to n test items in the test order. The m test units corresponding to the m machines, $m > n$, each test unit performs one kind of test item. Its goal is to find a schedule which minimizes finish time of the last job, denoted C_{max}.

Shifting bottleneck heuristic algorithm is used to solve deterministic job shop scheduling problem. MPTMSs can obtain exact test processing time from measurement systems, so scheduling unit is changed from day to hour. More precious processing time can be obtained by analyzing historical data about time dimension information of measurement data, designated test engineer and test unit. The processing times in Table III are average processing time of test unit belongs to different product type. Job1 and Job2 are the same product type with different test items and predefined sequences. Fig. 8 illustrates a feasible solution with $C_{max} = 500$ corresponding to the test data presented in Table III. The x axis in Fig. 8 represents machines listed in machine sequence column of Table III. The dispatch list interface of the solution is also given in Fig. 9, which shows the start time and the end time of each operation.

F. Process Mining

Server logs record the events triggered in workflow instances, which consists of events with workflow identification, event type, timestamp and originator information. Based on server logs, the patterns of workflow can be classified by included events. From originator information and timestamp information, the workload of each workflow participant can be got. Besides that, the bottleneck of those workflows can be calculated from timestamp information. By comparing those mined workflow patterns, futher information can be found, such as the reason of a longer pattern consumes less time than a short pattern, which may indicates deficiencies.

V. Application Example

In order to validate the application development framework, it is applied to develop washing machine testing management system. Fig. 10 exhibits washing machine test laboratory. The MPTMS and measurement system are showed in Fig.11 and Fig.12.

Figure 10. Washing machine test laboratory

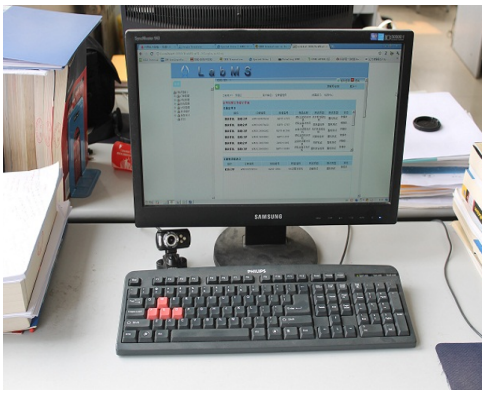

Figure 11. Management system

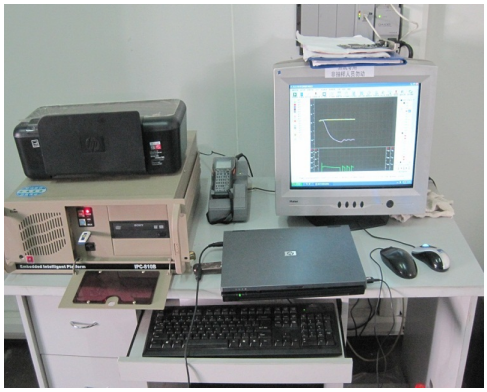

Figure 12. Measurement system

During the procedure of user requirement analysis, workflow model is designed. Fig. 13 and Fig. 14 demonstrates the workflow model. The process definition XML excerpt is given in Fig.15. Compared to the generic measurement-based product testing process, this process is more complicated and customized by user requirements. Messages are used to partition the complex process and decouple the branches process with the main process. The design of system is based on the workflow model. The workflow model is decomposed into flow objects. User interfaces, miscellaneous database tables for each flow object are further designed. Data forms and user interfaces are built using JSP. Java message service [30] is used to support the message handling. Workflow is driven by calling execution API exposed by workflow runtime environment. Measurement systems that comply with IEEE 1851 standard and test order information web

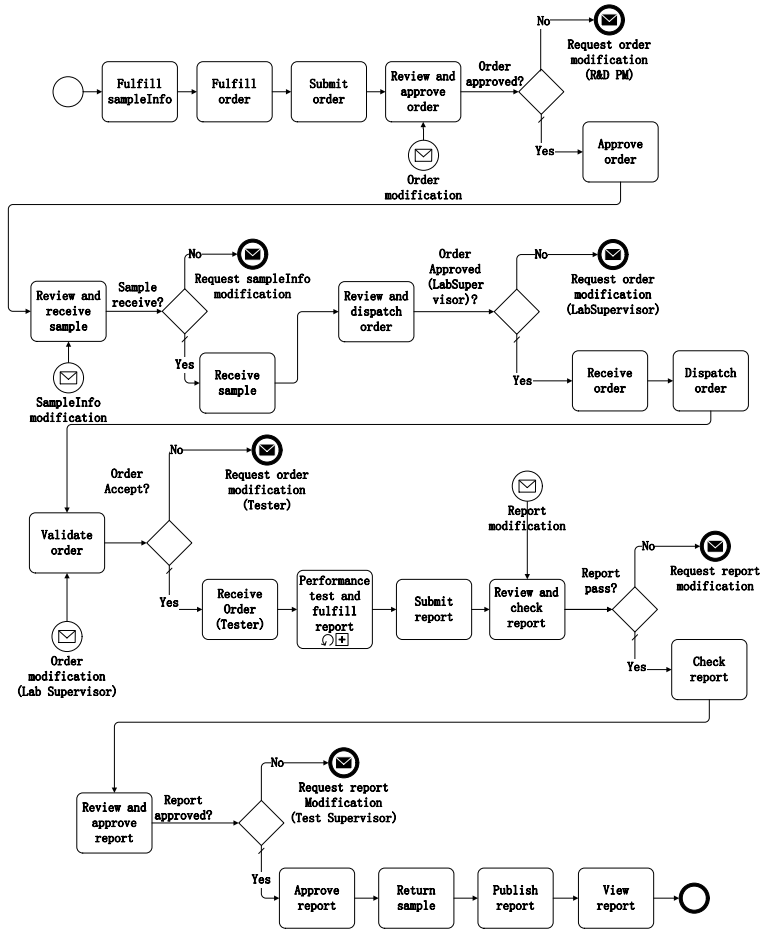

Figure 13. Workflow model for washing machine test laboratory

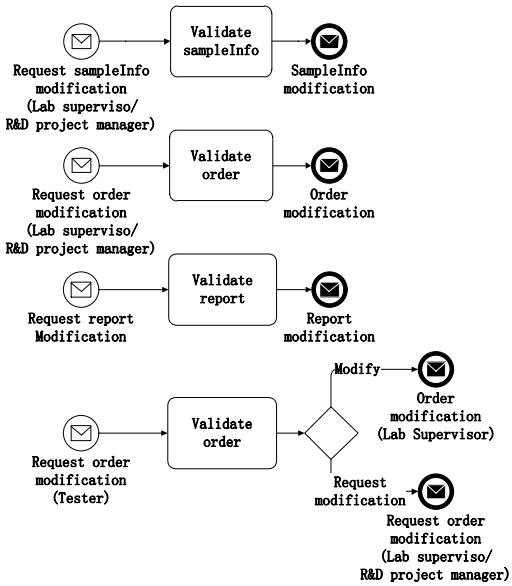

Figure 14. Message handling in workflow model for washing machine test laboratory

service interface can seamlessly integrate with MPTMSs.

Forty one cases are selected as example of measurement-based washing machine product testing process data. All those cases are completed and integrated cases translated from server logs. Each case consists of events with event type, timestamp and originator information. As depicted in Fig. 16, forty one cases are classified into twelve patterns which are ordered by frequencies and indicated by different colors. Activities between *SubmitOrder* and *Message:OrderModification* cost a large amount of time and frequently switch before entering the following activities. Along with roles assigned with those activities, it can be inferred that R&D engineers are not familiar with test standards, sample main parts and sample technical parameters. Lab supervisor had to send back the incorrect orders to the R&D engineers and wait for the R&D engineers correct and re-submit the orders. As depicted in Fig. 17, pattern1 has the maximum frequency of fifteen and second minimum average cost time, which indicates that most of the workflows go through the normal way and avoid the time spent on handling correcting errors in orders and reports. In Fig. 16, pattern3 is more likely cost time than pattern2. However in Fig. 17, pattern2 cost more time than pattern3. This contradiction can be explained by further exploring cases contained in pattern2. Using basic log statistics function of ProM, the overall duration of each processes contained in pattern2 can be found. In Fig. 18, process6347 and process6143 have the duration 959 hours that much higher than other processes. By further inspecting process6347, the reason of high durations is caused by R&D engineer's latency in *ViewReport* activity, which indicates that R&D

```
<?xml version="1.0" ?>
- <process name="modelExample"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="www.ouc.edu.cn/webNewTech/1.0"
    xsi:schemaLocation="www.ouc.edu.cn/webNewTech/1.0
    definitionSchema.xsd">
  + <start name="Start">
  + <activity name="Fulfill_sampleInfo">
  + <activity name="Fulfill_order">
  + <activity name="Submit_order">
  + <activity name="Review_and_approve_order">
  + <activity name="Arrove_order">
  + <activity name="Review_and_receive_sample">
  + <activity name="Receive_sample">
  + <activity name="Review_and_dispatch_order">
  + <activity name="Receive_order">
  + <activity name="Dispatch_order">
  + <activity name="Validate_order">
  + <activity name="Receive_order(Tester)">
  + <activity name="Perform_test_and_fulfill_report">
  + <activity name="Submit_report">
  - <activity name="Review_and_check_report">
      <transition name="to Report_pass" to="Report_pass" />
      <messageHandler name="RM_messageHandler" messageType="Rollback"
        messageContent="Report_modification" />
    </activity>
  + <activity name="Check_report">
  + <activity name="Review_and_approve_report">
  + <activity name="Arrpove_report">
  + <activity name="Return_sample">
  + <activity name="Publish_report">
  + <activity name="View_report">
  + <decision name="Report_approved">
  - <decision name="Report_pass">
      <transition name="to RRM_message" to="RRM_message" condition="No"
        type="message" />
      <transition name="to Check_report" to="Check_report" condition="Yes" />
      <message name="RRM_message" messageType="Rollback"
        messageContent="Request_report_modification" />
    </decision>
  + <decision name="Order_accpet">
  + <decision name="Order_approved_LabSupervisor">
  + <decision name="Sample_receive">
  + <decision name="Order_approved">
    <end name="End" />
  </process>
```

Figure 15. Process definition XML excerpt

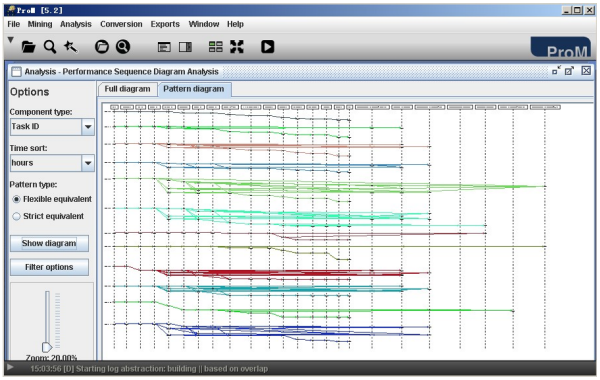

Figure 16. Pattern analysis using ProM

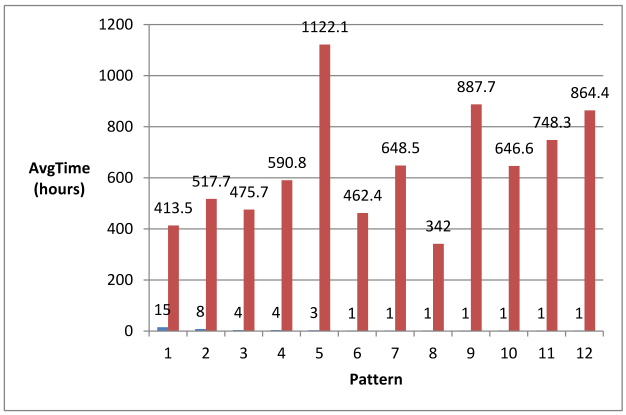

Figure 17. Pattern data with frequency and average cost time

engineers do not care about the order and forget to view the test report. By checking process6143, the reason of

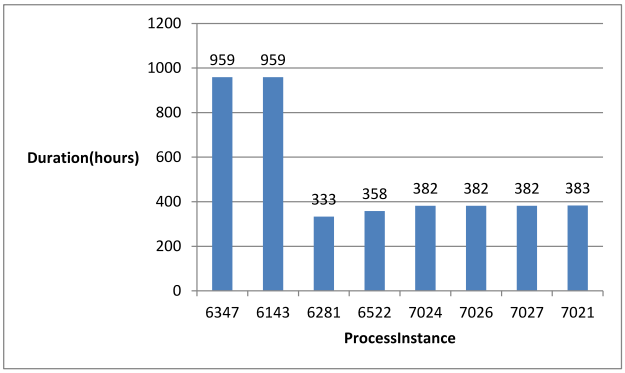

Figure 18. Processes contained in pattern two with durations

high duration is caused by Lab supervisor latency in *Message:Request order modification* activity which is eighteen days after *ReceiveSample* activity, which may indicates Lab supervisor does not notice this order due to bad user interface design.

VI. Conclusion

Based on the previous projects and user requirements, designs of MPTMSs are analyzed. Then, an application development framework for MPTMSs is presented. Followed by the application framework, the design of the framework is given. New MPTMS can be built by leveraging the exist components provided by the framework. IEEE 1851 standard interfaces and test order information web service interfaces are used to remove the integration barriers between measurement systems and MPTMSs. The development of MPTMS is complicated, so there are many limitations can be improved. For example, the application development framework depends on Apache Tomcat [31] web server and other J2EE [32] servers are not supported currently. The process mining of event logs can be given further study. Take into consideration of those limitations and growing requirements, the application development framework will be updated and extended to support efficient development of more complex MPTMSs in the future.

References

[1] P. Arpaia, L. Fiscarelli, G. La Commara, and C. Petrone, "A model-driven domain-specific scripting language for measurement-system frameworks," no. 99, pp. 1–11, 2011.

[2] L. Chen, "Automatic test cases generation for statechart specifications from semantics to algorithm," *Journal of Computers*, vol. 6, p. 4, 2011.

[3] J. Bosch and J. Bosch, "Design of an object-oriented framework for measurement systems," in *in Object-Oriented Application Frameworks*. John Wiley, 1998.

[4] A. Kalicki, L. Makowski, A. Michalski, and Z. Staroszczyk, "Instrumentationnotes - distributed measurement systems-a web system approach: Part i," *Instrumentation Measurement Magazine, IEEE*, vol. 11, no. 2, pp. 50 –56, april 2008.

[5] G. Lv, Z. Guo, S. Xie, and W. Pan, "Web-based real-time monitoring system on cold chain of blood," in *Instrumentation and Measurement Technology Conference, 2009. I2MTC'09. IEEE.* IEEE, 2009, pp. 1294–1299.

[6] Y. Y. Li Li, Jian Liu, "Research and development of intelligent motor test system," *Journal of Computers*, vol. 7, p. 9, 2012.
[7] A. Kalicki, L. Makowski, A. Michalski, and Z. Staroszczyk, "Instrumentationnotes - distributed measurement systems-a web system approach: Part 2," *Instrumentation Measurement Magazine, IEEE*, vol. 11, no. 6, pp. 44 –51, december 2008.
[8] J. Anderson, J.L., "How to produce better quality test software," *Instrumentation Measurement Magazine, IEEE*, vol. 8, no. 3, pp. 34 – 38, aug. 2005.
[9] A. Bondavalli, A. Ceccarelli, L. Falai, and M. Vadursi, "A new approach and a related tool for dependability measurements on distributed systems," *Instrumentation and Measurement, IEEE Transactions on*, vol. 59, no. 4, pp. 820 –831, april 2010.
[10] T. Z. J.-y. S. L.-l. S. JUN YANG, Bin Liang, "Laboratory test system design for star sensor performance evaluation," *Journal of Computers*, vol. 7, p. 4, 2012.
[11] M. Harvey, D. Scott, and P. Coveney, "An integrated instrument control and informatics system for combinatorial materials research," *Journal of chemical information and modeling*, vol. 46, no. 3, pp. 1026–1033, Apr. 2006.
[12] C. Cotofana, L. Ding, P. Shin, S. Tilak, T. Fountain, J. Eakins, and F. Vernon, "An soa-based framework for instrument management for large-scale observing systems (usarray case study)," in *Proceedings of the IEEE International Conference on Web Services*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 815–822.
[13] E. Deelman, G. Singh, M. hui Su, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, G. B. Berriman, J. Good, A. Laity, J. C. Jacob, and D. S. Katz, "Pegasus: a framework for mapping complex scientific workflows onto distributed systems," *SCIENTIFIC PROGRAMMING JOURNAL*, vol. 13, pp. 219–237, 2005.
[14] J. Cao, S. Jarvis, S. Saini, and G. Nudd, "Gridflow: Workflow management for grid computing," in *Cluster Computing and the Grid, 2003. Proceedings. CCGrid 2003. 3rd IEEE/ACM International Symposium on*. IEEE, 2003, pp. 198–205.
[15] S. Jablonski and C. Bussler, *Workflow management: modeling concepts, architecture and implementation*. International Thomson Computer Press, 1996.
[16] S. White, "Introduction to bpmn," *IBM Cooperation*, pp. 2008–029, 2004.
[17] Z. Guo, P. Chen, Y. Feng, Y. Jiang, and F. Hong, "Isdp: Interactive software development platform for household appliance testing industry," vol. 59, no. 5, pp. 1439–1452, 2010.
[18] D. Grimaldi and M. Marinov, "Distributed measurement systems," *Measurement*, vol. 30, no. 4, pp. 279–287, Dec. 2001.
[19] Z. Guo, P. Chen, H. Zhang, M. Jiang, and C. Li, "Ima: An integrated monitoring architecture with sensor networks," *Instrumentation and Measurement, IEEE Transactions on*, vol. 61, no. 5, pp. 1287 –1295, may 2012.
[20] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, *et al.*, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.
[21] W. Ren and X. Lin, "Realization of medicine warehouse management based on spring framework and extjs [j]," *Computer Engineering and Design*, vol. 18, p. 053, 2009.
[22] A. Sahuguet and F. Azavant, "Wysiwyg web wrapper factory (w4f)," 1999.
[23] YuYu, Z. Guo, Z. Sun, and Y. Lu, "An integrated application framework for measurement-based product testing management," in *Instrumentation and Measurement Technology Conference (I2MTC)*. IEEE, May. 2012.
[24] B. Van Dongen, A. de Medeiros, H. Verbeek, A. Weijters, and W. Van der Aalst, "The prom framework: A new era in process mining tool support," *Applications and Theory of Petri Nets 2005*, pp. 1105–1116, 2005.
[25] "Ieee standard for design criteria of integrated sensor-based test applications for household appliances," *IEEE P1851/D1.3, April 2012*, pp. 1 –53, 4 2012.
[26] M. Zur Muehlen, "Organizational management in workflow applications - issues and perspectives," *Inf. Technol. and Management*, vol. 5, pp. 271–291, Jul. 2004.
[27] K. Trichkov and E. Trichkova, "Modeling and execution of web service in internet using enhydra workflow platform," in *International Conference on Computer Systems and Technologies-CompSysTech*, vol. 6, 2006.
[28] M. Cumberlidge, "Business process management with jboss jbpm: A practical guide for business analysts," 2007.
[29] M. Pinedo, *Scheduling: theory, algorithms, and systems*. Springer Verlag, 2008.
[30] M. Hapner, R. Burridge, R. Sharma, J. Fialli, and K. Stout, "Java message service," *Sun Microsystems Inc., Santa Clara, CA*, 2002.
[31] V. Chopra, S. Li, and J. Genender, *Professional Apache Tomcat 6*. Wrox, 2011.
[32] D. Alur, J. Crupi, and D. Malks, *Core J2EE patterns: best practices and design strategies*. Prentice Hall PTR, 2003.

Zhongwen Guo was born in China in 1965. He received the B.S. degree in computer science and technology from Tongji University, Shanghai, China, in 1987 and the M.S. and Ph.D. degrees from Ocean University of China, Qingdao, China.

He is currently a Professor and Doctoral Advisor with the Department of Computer Science and Engineer, Ocean University of China. His main research interests focus on sensor networks, distributed measurement systems, ocean monitoring, and so on.

Yu Yu was born in China in 1984. He received the M.S. degree in computer science and techology from Shandong University of Science and Technology in 2010.

He is currently working toward the Ph.D. degree with the Department of Computer Science and Technology, Ocean University of China, Qingdao, China, where he is researching information management system.

Zhaosui Sun received the B.S. degree in Qufu normal university, Qufu, China, in 1987 and the M.S. degree from Tianjin University, Tianjin, China in 1992.

He is currently working toward the Ph.D degree with the Department of Computer Science and Technology, Ocean University of China, Qingdao, China, where he is researching distributed management system.

Yunhong Lu was born in China in 1972. He received the B.S. degree in Yantai university, Yantai, China, in 1994 and the M.S. degree from Jilin university, Jilin, China in 2001.

He is currently working toward the Ph.D. degree with the Department of Computer Science and Technology, Ocean University of China, Qingdao, China, where he is researching wireless sensor network.