Object-Oriented Programming Applied to Laboratory Automation. 3. The Standard Robot Interface Protocol for the Analytical Director

T. Zhou and T. L. Isenhour*

Department of Chemistry, Kansas State University, Manhattan, Kansas 66506

J. C. Marshall

Department of Chemistry, Saint Olaf College, Northfield, Minnesota 55057

Received June 1, 1992[⊗]

A structured robot interface protocol has been developed for the Analytical Director. The fundamental automation capability of a robotics laboratory is defined by a standard set of intrinsic functions and programmable control commands at a low level. Analytical procedures are represented and saved in chemistry notation format using a set of standard syntax and semantic rules. With the robot interface protocol, analytical procedures in chemistry notation format can be translated by the Analytical Director to low level robot commands and executed by different robot laboratories. The object-oriented modeling of an analytical laboratory and the four-layer structured robot interface protocol are discussed in this paper. The robot execution of an analytical method is also presented and compared with human expert analyses.

The Analytical Director is a project with the goal of combining artificial intelligence and laboratory robotics into a system that can design, simulate, test, and implement its own analytical laboratory procedures. Figure 1 shows the architecture of the Analytical Director, which consists of *user interface*, *conceptual model of laboratory*, and *robotic table*. A conceptual laboratory model[1,2] has been developed using object-oriented design techniques to simulate a robotic analytical laboratory. A user designs an analytical procedure using chemistry knowledge at the conceptual level. The conceptual laboratory model accepts the user input and represents an analytical procedure by a set of objects. The conceptual laboratory model validates the designed procedure and saves the conceptual procedure in chemistry notation format. The robot controller receives a sequence of robot commands from the conceptual laboratory model and controls the robot execution necessary to accomplish the analysis.

With the Analytical Director, analytical procedures can be designed and saved at a conceptual level using chemistry notation (or pseudolanguage) in one laboratory, transferred to other robotic laboratories, and executed there with same quality assurance. A conceptual procedure consists of a set of objects which contain the knowledge of abstract laboratory operations and the associated chemical reactions. Such a procure is independent of the robotic laboratory equipment. The robot execution of this *conceptual procedure* is accomplished by converting the conceptual procedure into a sequence of robot commands which are dependent on robotic laboratory equipment. To assure different robotic laboratories execute a conceptual procedure with the same quality, a standard robot interface protocol must be defined to deal with the diverse robotic laboratory equipment.

In this paper, we will first discuss briefly the object-oriented design of the conceptual laboratory model. Then we will demonstrate how an analytical procedure can be converted to a robot executable procedure. Finally, we will discuss the standard robot interface protocol and its implementation.

CONCEPTUAL MODEL OF ROBOTIC ANALYTICAL LABORATORY

A conceptual model of a robotic analytical laboratory must capture the full range of knowledge required for a computer system to design, plan, and ultimately control the robot actions during the execution of laboratory procedures. The conceptual model developed includes

- an abstract state space representation of the robotic table: It consists of state variables whose values are chemical reagents, samples, standard solutions, test tubes, laboratory instruments and devices, robot hands, etc. This state space representation will be referred as the *workbench*.
- an abstract representation of the primitive steps of an analytical procedure: A new state may be generated by applying a primitive procedure step to a state of the workbench.
- an abstract representation of each analytical procedure: It consists of a legal sequence of primitive procedure steps.
- a chemical information system[3] that provides a strategy for a deep-reasoning based knowledge inference engine for generating a legal sequence of primitive procedure steps.

An object-oriented design technique is employed in developing the conceptual laboratory model. An object incorporates both data structure (data attributes) and behavior (method attributes).[4] Each object has "intelligence": it knows its own state, represented by data attributes, and can operate on and alter its state by activating method attributes.

The abstract *workbench* consists of a set of objects as a computer representation of the laboratory physical resources. Objects representing laboratory equipment include a balance, a water bath, a spectrophotometer, a centrifuge, a pH meter, a mixer, stocks, test tubes, and the robot. Stocks are used for storing chemical reagents, samples, and standard solutions. Test tubes are used for performing primitive operations, such as mixing, centrifuging, separating, and diluting.

Figure 2 shows the stock object definition. The data attributes specifies the volume, name, state, and chemical components of a stock solution. The method attributes specify

[⊗] Abstract published in *Advance ACS Abstracts*, March 15, 1994.

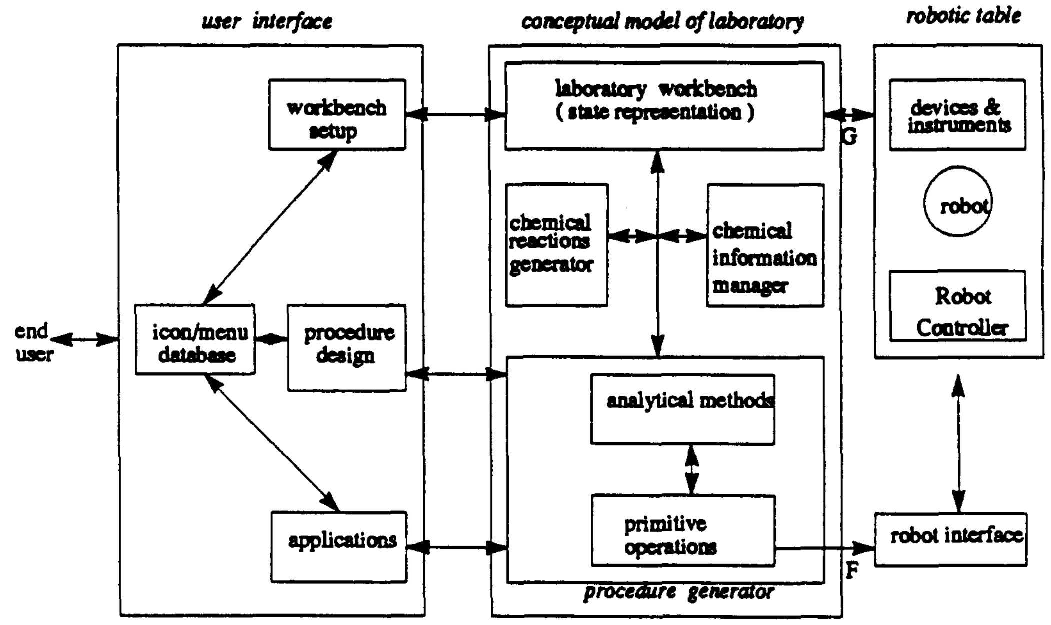
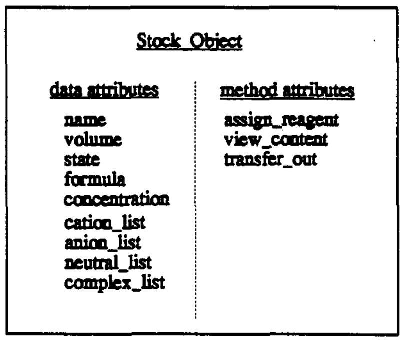


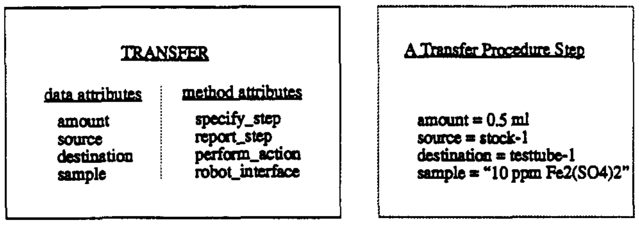
Figure 1. Architecture of the Analytical Director.



Stock Object

data attributes	method attributes
name	assign_reagent
volume	view_content
state	transfer_out
formula	
concentration	
cation_list	
anion_list	
neutral_list	
complex_list	

Figure 2. Definition of the stock object.



TRANSFER

data attributes	method attributes
amount	specify_step
source	report_step
destination	perform_action
sample	robot_interface

A Transfer Procedure Step

amount = 0.5 ml
source = stock-1
destination = testtube-1
sample = "10 ppm Fe2(SO4)2"

Figure 3. Object definition for transfer primitive operation.

the behavior of a stock object. The *assign_reagent* is a method attribute for assigning new stock solution. The *view_content* is a method for displaying a description of the stock solution. The *transfer_out* is a method for transferring a certain amount of solution from a stock to a test tube.

Shank and his colleagues[5] maintain that a small set of primitive actions will account for what must be represented in the physical world. Following this philosophy, we have grouped the activity of the standard laboratory robotics system into families of primitive actions. We list below the families of primitive actions necessary for *volumetric*, *gravimetric*, and *spectrophotometric* analyses.

ADD	SELECT.WAVELENGTH
CENTRIFUGE	SEPARATE
CONTROL.PH	STIR
CONTROL.TEMPERATURE	TITRATE
DILUTE	TRANSFER
MEASURE.ABSORBANCE	WAIT
MEASURE.PH	WEIGHT

Each of these primitive operations is represented by an object. Each of these objects has data attributes and method attributes. The data attributes specify the parameters about the operation. The method attributes specify how the operation will be simulated in the conceptual model to change the state space of the workbench. Each procedure step is an instance of one of these primitive operation objects. Figure 3 shows the definition of the TRANSFER primitive operation and the representation of a procedure step "transferring 0.5 mL of 10 ppm of $Fe_2(SO_4)_2$ from stock-1 to testtube-1".

A designed procedure in the conceptual model consists of a set of objects, each representing a procedure step. This set of objects is referred as the conceptual procedure.

CONVERTING A STANDARD WRITTEN TEXT PROCEDURE TO A STANDARD ROBOT PROCEDURE

This section presents how a standard written text procedure can be converted to a standard robot procedure. The phenanthroline method for determining iron concentration[6] is used to demonstrate the system performance. Iron is reduced to the ferrous state with hydroxylamine and treated with 1,-10-phenanthroline at pH 3.2–3.3. The same text procedure was performed by the *Chemical Analysis Lab* class for the 1992 spring semester at Kansas State University. Comparisons on the time and analyses results are made at the end of this section.

Chemicals and the Standard Written Procedure. Reagents. All chemicals used were reagent grade or the best quality commercially available. All the standards and reagent solutions are prepared using distilled and deionized water.

Hydrochloric acid

6 N HCl, containing less than 0.000 05% iron

Hydroxylamine solution

dissolve 10 g of $NH_2OH{\cdot}HCl$ in 100 mL of water

Ammonium acetate buffer solution

dissolve 250 g of $NH_4C_2H_3O_2{\cdot}H_2O$ in 150 mL of water; add 700 mL of concentrated (glacial) acetic acid; because even a good grade of $NH_4C_2H_3O_2{\cdot}H_2O$ contains a significant amount of iron, prepare new reference standards with each buffer preparation

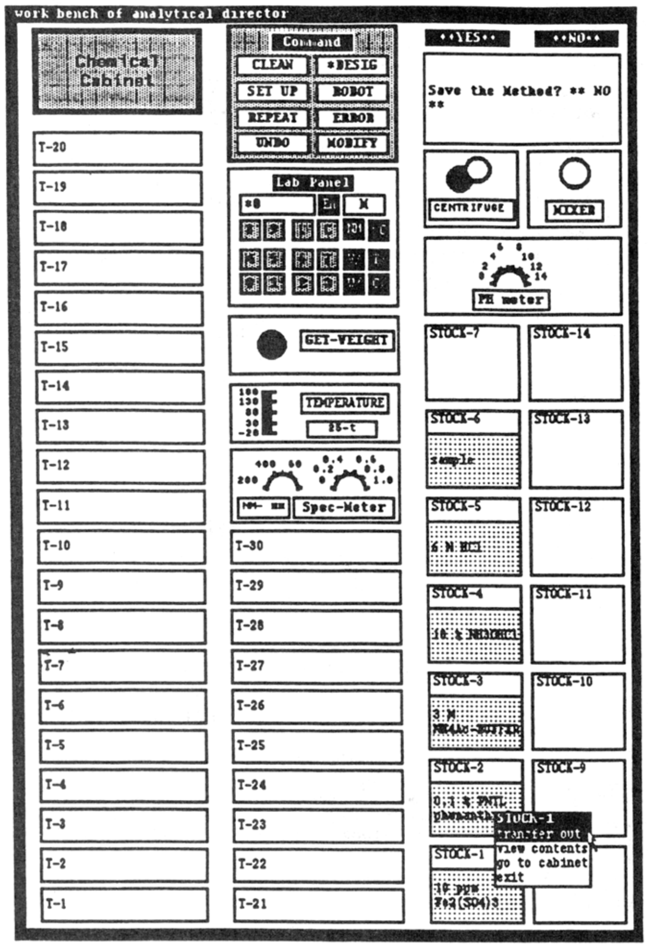

Save the Method? ** NO **

Figure 4. Graphical representation of the workbench.

Phenanthroline solution
dissolve 100 mg of 1,10-phenanthroline monohydrate, $C_{12}H_8N_2{\cdot}H_2O$, in 100 mL of water by stirring and heating to 80 °C

Stock iron solution
slowly add 20 mL of concentrated H_2SO_4 to 50 mL of water and dissolve 1.404 g of $Fe(NH_4)_2(SO_4)_2{\cdot}6H_2O$; add 0.1 N potassium permanganate ($KMnO_4$) dropwise until a faint pink color persists; dilute to 1000 mL with water and mix; 1.00 mL = 200 μg of Fe

Standard iron solution
prepare daily for use

Procedure. Into six 100-mL erlenmeyer flasks, pipet 0- (blank), 5-, 10-, 15-, 20-, and 25-mL portions of standard iron solution. Into another two 100-mL erlenmeyer flasks, pipet 20 mL of unknown sample. Add 2 mL of 6 N HCl and 1 mL of $NH_2OH{\cdot}H_2O$ solution. Place the erlenmeyer flasks into a boiling water bath for 10 min. Cool to room temperature, and transfer to a 100-mL volumetric flask. Add 10 mL of $NH_4C_2H_3O_2$ buffer solution and 4 mL of phenanthroline solution, and dilute to mark with water. Mix thoroughly and allow at least 10–15 min for maximum color development.

Designing the Analytical Procedure at the Conceptual Level. An icon based user interface has been developed to enable a user to interact with the conceptual laboratory model to design analytical procedures. Figure 4 shows the iconic representation of the major components of workbench, which includes 30 test tubes, 14 stock containers, a chemical cabinet, a command box, a laboratory panel box, a balance, a water bath, a spectrophotometer, a centrifuge, a pH meter, a YES box, a NO box, and a PROMPT window. A previous paper has presented the design and implementation of the icon-based user interface.[2] An icon is a graphical representation of an object created in the conceptual model. A user can use a mouse to click on an icon, and the icon responds to a mouse click by popping up a menu. Each menu contains a list of applicable design activities. A user operates on these icons and menus to design an analytical procedure. For example, a mouse clicking on the empty *stock-1* icon would allow a user to assign a new chemical reagent to the stock-1 object. Designing an analytical procedure in the conceptual laboratory model consists of two phases: (1) assigning chemical reagents to stock solutions; (2) designing each step of a procedure.

To assign a new chemical reagent to a stock solution, a user can use a mouse to click on an empty stock icon and choose the "assign new reagent" menu item (refer to Figure 4). The system will then prompt the user to input the chemical formula (or chemical name) of the reagent, and the concentration of the reagent if it is a solution. Figure 4 shows the graphical display of the workbench after all the reagents for the method are assigned.

A conceptual procedure captures the operational principle and the associated chemical reactions of a method. The volume of the standard written procedure is down scaled to reflect the fact that a test tube can hold up to 10 mL of solution. One test tube (testtube-21 for this example) is used to design the procedure. Because the usage of test tubes for the robot execution is scheduled by the procedure generator of the conceptual model, it is not important which test tube (testtube-1 or testtube-21) to be used to represent the conceptual procedure. Figure 5 shows the icon manipulations of designing each step of the procedure.

The first step is to transfer 2 mL of iron solution from stock-1 to testtube-21. The user clicks on the stock-1 icon. A menu with all the possible operations for stock-1 pops up, from which the user chooses "transfer out". The user is then prompted to specify the amount (2 mL) being transferred and the destination (testtube-21).

To design the second step, the user clicks on the *stock-5* icon and selects the option transfer_out from the pop up menu. This selection activates the transfer_out method of the stock-1 object which prompts the user to specify the amount to be transferred (0.2 mL) and the destination (testtube-21).

As shown in Figure 5, the user simulates the laboratory activities by operating on icons to design a conceptual procedure. When a relevant icon is selected, the corresponding object is activated which pops up the possible operations and assists the user to design a procedure step.

During the process of designing a procedure, relevant chemical reactions are simulated by the conceptual model to change the chemical components of a test tube. Notice that the chemical state change of the iron from $Fe_2(SO_4)_2$ (step 2) to the Fe^{II}–PNTL complex (step 7). Such chemical reaction simulation is used to construct a deep-reasoning based knowledge inference engine which assists the designing process and validates a designed procedure.

The designed procedure is saved in two formats, as object specification and English description. In the object specification format, each stock object or procedure step object is saved as a verbal description. Figure 6 shows part of the phenanthroline procedure in the object specification: the standard iron stock solution and the first and last procedure steps. The conceptual model can read a procedure in object specification format and generate the corresponding procedure objects in the conceptual model.

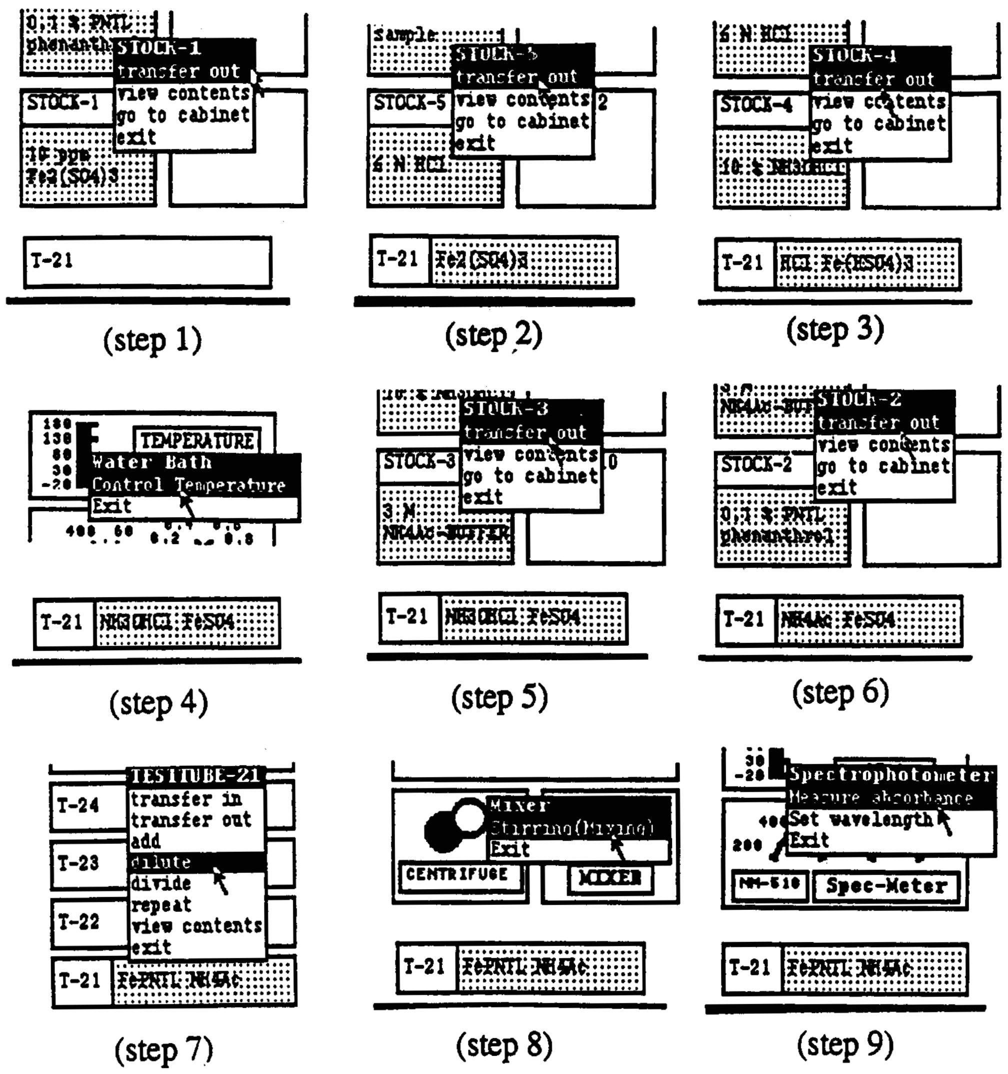


Figure 5. Icon manipulations for designing a conceptual procedure.

```
STOCK-1

name = "Ferric sulfate"
volume = 250 ml
state = solution
formula = "Fe2(SO4)2"
concentration = 10 ppm
molar_concentration = 1.79E-4
cation_list = (("Fe" C 3))
anion_list = (("SO4" A -2"))
```

```
step-1: TRANSFER

amount = 0.5 ml
source = stock-1
destination = testtube-21
sample = "10 ppm Fe2(SO4)2"
```

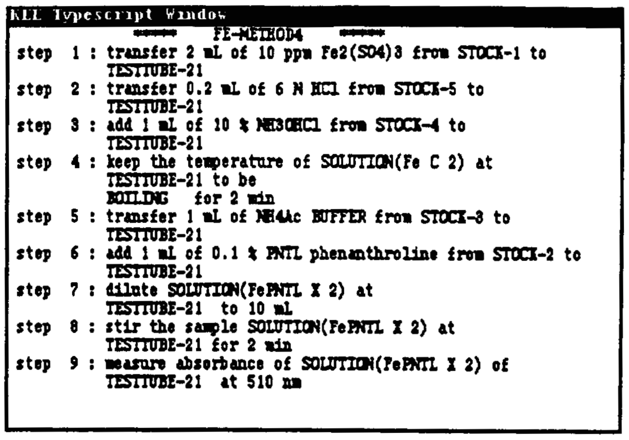
```
step-9: MEASURE.ABSORBANCE

wavelength = 520 nm
source = stock-1
destination = testtube-21
sample = "solution(FePNTL x 2)"
```

Figure 6. Procedure saved in object specification format.

Figure 7 shows the procedure in English description format. A procedure in English description is easier to read by a human expert. The developed expert system applies a set of syntax and semantic rules along with the system knowledge base to convert such an English description to the conceptual procedure objects.

Executing a Designed Procedure. The conceptual procedure underlines the chemical principle of an analytical method. The real robot procedures for running different samples and using different standards is generated by the system each time an application needs to be performed. Figure 8 shows a typical way a user starts the robotic laboratory to execute a designed procedure. After the user specifies the target element *iron*, the system pops up a menu containing the current available procedures for that element. Four methods have been designed for iron analysis. Method 4 is selected to perform the robot analysis. Before the robot laboratory executes a conceptual



```
KEE Typescript Window
            ─────   FE-METHOD4   ─────
step  1 : transfer 2 mL of 10 ppm Fe2(SO4)3 from STOCK-1 to
          TESTTUBE-21
step  2 : transfer 0.2 mL of 6 N HCl from STOCK-5 to
          TESTTUBE-21
step  3 : add 1 mL of 10 % NH3OHCl from STOCK-4 to
          TESTTUBE-21
step  4 : keep the temperature of SOLUTION(Fe C 2) at
          TESTTUBE-21 to be
          BOILING   for 2 min
step  5 : transfer 1 mL of NH4Ac BUFFER from STOCK-3 to
          TESTTUBE-21
step  6 : add 1 mL of 0.1 % PNTL phenanthroline from STOCK-2 to
          TESTTUBE-21
step  7 : dilute SOLUTION(FePNTL X 2) at
          TESTTUBE-21  to 10 mL
step  8 : stir the sample SOLUTION(FePNTL X 2) at
          TESTTUBE-21 for 2 min
step  9 : measure absorbance of SOLUTION(FePNTL X 2) of
          TESTTUBE-21  at 510 nm
```

Figure 7. Procedure saved in English description format.

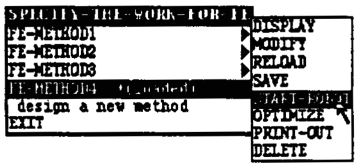
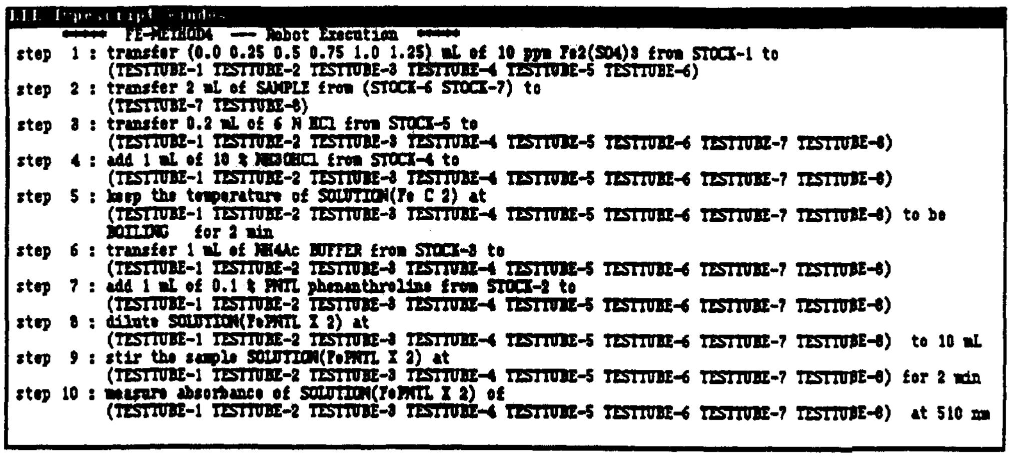


Figure 8. Start the robot execution of a conceptual procedure.

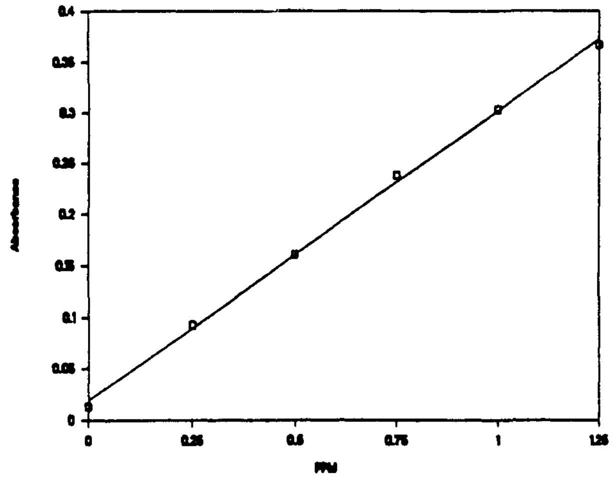
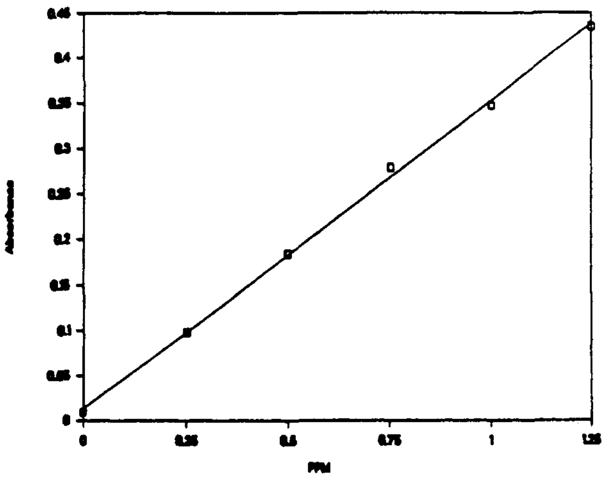
procedure, the system prompts the user to specify the number of samples, location of samples, standard concentration, etc. Suppose one particular application needs to run two samples with a 10 ppm iron standard. The user specifies the samples



```
 ------ FE-METHOD4  --- Robot Execution ------
step  1 : transfer (0.0 0.25 0.5 0.75 1.0 1.25) mL of 10 ppm Fe2(SO4)3 from STOCK-1 to
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6)
step  2 : transfer 2 mL of SAMPLE from (STOCK-6 STOCK-7) to
          (TESTTUBE-7 TESTTUBE-8)
step  3 : transfer 0.2 mL of 6 N HCl from STOCK-5 to
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8)
step  4 : add 1 mL of 10 % NH3OHCl from STOCK-4 to
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8)
step  5 : keep the temperature of SOLUTION(Fe C 2) at
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8) to be
          BOILING   for 2 min
step  6 : transfer 1 mL of NH4Ac BUFFER from STOCK-3 to
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8)
step  7 : add 1 mL of 0.1 % PNTL phenanthroline from STOCK-2 to
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8)
step  8 : dilute SOLUTION(FePNTL X 2) at
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8)  to 10 mL
step  9 : stir the sample SOLUTION(FePNTL X 2) at
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8) for 2 min
step 10 : measure absorbance of SOLUTION(FePNTL X 2) of
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8)  at 510 nm
```

Figure 9. Complete robot procedure generated in the conceptual model.

Table 1. Comparison between Robot Laboratory and Human Expert Executions

	human experts	robot laboratory
	Time	
students	2 h 18 min ± 22 min (in pairs)	1 h 47 min ± 0
TAs[a]	2 h 31 min ± 12 min	
		
	Concentration of Sample-1	
five trials	0.635, 0.634, 0.569, 0.601, 0.612	0.602, 0.617, 0.618, 0.618, 0.587
	0.610	0.608
95% confidence level	0.610 ± 0.034	0.608 ± 0.017

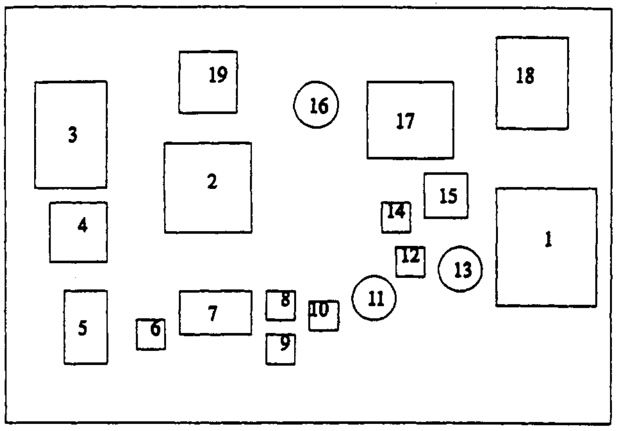
[a] TAs = teaching assistants.

to be at *stock-6* and *stock-7*. The standard is specified to be at *stock-1* with a concentration of 10 ppm. The user also specifies not to use a standard addition method for there is no significant matrix effect.

The *Procedure Generator* (PG) utilizes the application information and the designed conceptual procedure along with the system knowledge base to generate a complete robot procedure. Figure 9 shows the English description of a complete robot procedure for analyzing two samples. It is important to notice that both the conceptual procedure and the robot procedure are actually represented in the conceptual model by objects.

Results and Discussion. Three sets of data are collected, the robot execution, the analyses by three teaching assistants, and the results from 72 students. There were five sections for the chemical analysis class of 1992 Spring semester. The robot laboratory executed the same procedure when students of each section started their experiment. Students performed the analyses in pairs of two. Thirty six sets of data from students were collected. Once started, the robot execution was conducted without human being intervention, and data were automatically saved by the system. In order to compare the robot execution with the analysis by human experts, three teaching assistants volunteered to perform the same experiment. Table 1 summarizes the results.

Performing the analyses in pairs of two, students spent an average of 2 h and 18 min (with a standard deviation of 22 min) to finish an experiment. Three TAs (teaching assistants) used an average of 2 h and 31 min (with a standard deviation of 12 min). The robot finished the analyses in 1 h and 47 min. Designing a procedure in the conceptual model only involves chemistry knowledge and icon-based computer operation skills. A chemist capable of using the icon-based user interface would take 10–20 min to convert such a standard written text procedure into the conceptual procedure. The standard working curve was plotted to show the concentration–absorbance linear relationship at certain concentration range. While there were some noticeable quality differences among the three standard working curves of the TAs, the five standard working curves from the robot execution showed almost



(1) Zymate laboratory controler equipped with a Z-840 computer interface
(2) Zymate II Laboratory Robot
(3) Heulett-Packard 8451 A diode array spectrophotometer
(4) Stock rack
(5) Mettler AE 100 balance equipped with a Z-850 balance interface
(6) Zymate general purpose hand
(7) Zymate test tube rack
(8) Zymate syringe tip rack
(9) Zymate syringe hand for liquid distribution
(10) Zymate cannula hand for liquid distribution
(11) contact switch station
(12) Zymate liquid dispensing station
(13) waste dispensing station
(14) Zymate voltex station
(15) Zymate master laboratory station
(16) Zymate capping station
(17) Zymate centrifuge
(18) Zymate power and event control station
(19) S/P water bath

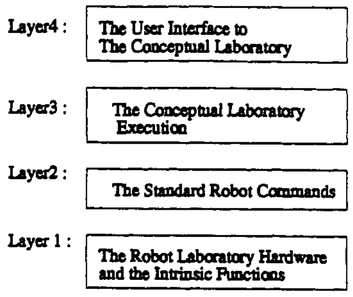
Figure 10. Layout of the robot laboratory hardware configuration.

identical linear relationships. Table 1 shows the best standard working curve of the three TAs and a typical one from the robot's execution. Five results of the determined concentration of sample-1 both from three TAs and five robot's results are compared in Table 1. The human analyses gave an average of 0.610 with a standard deviation of 0.027. The robot execution was faster, was automated, and gave an average of 0.608 with a standard deviation of 0.014. The robot laboratory can keep the experimental conditions (time for controlling the temperature at the water bath, time for color development, etc.) almost identical for different sessions. It would be hard for different human experts to control these experimental conditions that well.

THE STANDARD ROBOT INTERFACE PROTOCOL

The Robotic Laboratories. An intelligent robotic laboratory system consists of a robotic laboratory hardware configuration and a knowledge-based computer software package. Figure 10 shows the layout of our current robot laboratory hardware configuration. A Zymate laboratory automation system (Zymark Corp., Hopkinton, MA) was interfaced with a diode array spectrophotometer, a balance, and a pH meter. The robotic laboratory is capable of performing three classical analytical methods, *spectrophotometric*, *gravimetric*, and *volumetric*. The system controller runs the Zymate System V Software to control the robot activities and other instruments operations. Knowledge-based computer systems have been developed on both a SUN workstation and an IBM 80386-SX personal computer. The expert system supervision is achieved by interfacing the robot controller either with the SUN workstation or with the personal computer.

Both the SUN workstation and the personal computer-based systems adopt knowledge-based approach to combined artificial intelligence (AI) technique with robotic systems.



Layer4 :
The User Interface to The Conceptual Laboratory
Layer3 :
The Conceptual Laboratory Execution
Layer2 :
The Standard Robot Commands
Layer 1 :
The Robot Laboratory Hardware and the Intrinsic Functions

Figure 11. Four-layer structured robot interface protocol.

The SUN workstation-based computer system is developed under KEE. KEE is a knowledge engineering environment developed by IntelliCorp for expert system applications. Analytical procedures can be designed, modified, and executed under the expert system supervision. The analytical procedures are saved in two formats, *conceptual description* and *objects specification*. The PC-based computer system has been developed using BORLAND C++ and a Windows environment. Analytical procedures designed in one system can be delivered to the other system and executed with the same outcome. This is limited to the *object specification* because the chemical information module on the PC has not been developed yet. The purpose of the research on the standard robot interface protocol is to design and implement a multilayer structured robot interface so that analytical procedures designed in the conceptual laboratory model can be executed in different robotic laboratories reproducibly.

Figure 11 shows the four-layer structure of the developed robot interface. The first layer specifies the robotic laboratory physical resources and their fundamental automation functions. The second layer defines a set of standard robot commands. The third layer deals with the translation from a conceptual procedure to a sequence of robot commands. The fourth layer defines the user interface to the conceptual model for natural language or pseudo chemistry language processing.

Layer 1: The Physical Resources and The Intrinsic Functions. The physical resource layer specifies the hardware configuration of a robot laboratory and its automation capability. The current robot laboratory hardware configuration includes two types of containers, stocks and test tubes. The stock rack holds up to 15 250-mL polypropylene round stock bottles. The test tube rack holds up to 50 20 × 125 mm Pyrex test tubes. For spectrophotometric determinations, the test tubes are selected to assure their absorbance difference (in visible absorption) within the range of 0.015. The robot is equipped with three hands, general purpose hand, syringe hand, and cannula hand. The general purpose hand is used to move test tubes around. The legal positions a test tube can be moved to include the spectrophotometer cell holder for absorbance measurement, the water bath rack for temperature control, the centrifuge station for centrifuging, the votex station for stirring, the pH meter for pH measurement, and the test tube rack itself. The syringe hand is used to transfer less than 1 mL of solution from a stock to a test tube. A syringe tip is needed by the syringe hand to transfer solution. The cannula hand along with the liquid dispensing station is used to transfer more than 1 mL of solution from a stock to a test tube. The cannula hand can also be used to transfer solution from a test tube to another test tube.

The layer 1 robot programming defines a set of functions characterizing fundamental capabilities of each instrument and device. This set of functions will be referred to as the robot laboratory intrinsic functions. Table 1 lists the complete intrinsic functions for the current robot hardware configuration. The intrinsic functions have the following characteristics:

Intrinsic functions are the most fundamental robot unit operations. An intrinsic function is either a sequence of robot arm movements or a simple control statement of an instrument (or device). Consider the four intrinsic functions associated with the vortex station, *Above_Votex*, *IntoVotex*, *VotexOn*, and *VotexOff*. The Above_Votex moves a test tube (grabbed by the general purpose hand) to a position above the votex station. The IntoVotex moves a test tube into the votex holder. The VotexOn is a device control statement to turn on the stirrer. The VotexOff intrinsic function is a statement to turn off the stirrer. The layer 1 programming simplifies the complexity of robot laboratory control by treating each individual equipment in an isolated manner and associating each intrinsic function with several pre-existing conditions. For example, the Above_Votex intrinsic function has two preconditions. It assumes that the robot arm has attached the general purpose hand. It also assumes that the robot hand has already picked up a test tube. The mechanism for checking preconditions and scheduling execution sequence is implemented in layer 2 programming. This simplifies the layer 1 programming: intrinsic functions about a particular piece of equipment can be defined and updated without affecting other equipment and their intrinsic functions.

Intrinsic functions directly correspond to the capabilities of an instrument or a device. The automation capability of an instrument or a device is specified by several intrinsic functions. Take the spectrophotometer as an example; the HP 8451A diode array spectrophotometer has an integrated BASIC language programming capability. With an appropriate program, the spectrophotometer can accept a command from a remote computer to measure absorbance at a given wavelength. The spectrophotometer can also be programmed to scan a sample at a certain range of wavelength. Thus, two intrinsic functions *MeasureAbs*(*w*) and *Scan*(*from to*) are defined.

Intrinsic functions are equipment specific. Instruments with different manufacturers or different models may take different software control to accomplish the same operations. A different robot laboratory configuration requires a different set of intrinsic functions.

Intrinsic functions are equipment layout specific. This implies that some intrinsic functions need to be redefined if an instrument or a device is placed at a different position. With the robot hand rotation as an exception, the robot arm movement is accomplished with a sequence of three-dimensional positions along with the robot arm moving speed. There are two methods in defining each position, absolute and relative. An absolute position is specified with the robot arm rotation angle, height, and its reach. A relative position is defined by an one-dimensional parameter relative to the previous position. This parameter can be either moving downward, or moving upward, or rotating clockwise, etc. The function *AboveSpec*() is an example of moving the robot arm to an absolute position, right above the spectrophotometer cell holder. The function *IntoSpec*() is an example of moving the robot arm downward by certain distance relative to the previous position. Only the sequence IntoSpec() following the AboveSpec() can place a test tube into the cell holder of the spectrophotometer.

Intrinsic functions include no movement error checking. These functions guarantee robot movement error free if the precondition is satisfied. But intrinsic functions include no error checking mechanism, robot movement error is inevitable if an intrinsic function is called without the precondition satisfied. For example, a command *AttachCan* following *AttachGP* will cause the robot arm to first attach the general purpose hand and then try to attach the cannula hand—a collision error.

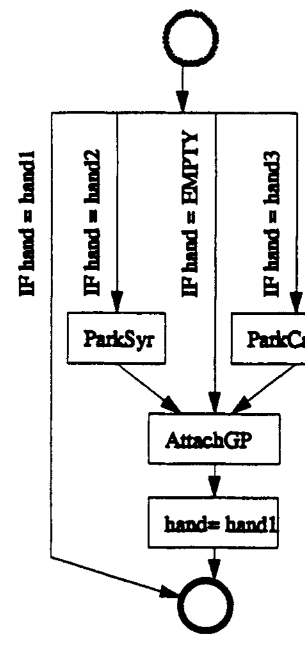


Figure 12. Structure of the GetHand1 robot command.

Intrinsic functions include no scheduling mechanism for logical error checking. Intrinsic functions are defined in an isolated manner to achieve modularity of the layer 1 programming. But an intrinsic function cannot accomplish the operational purpose unless called within a sequence of other intrinsic functions. Consider the intrinsic function *PullSyringe*(*amount*) of the liquid dispensing station. It is supposed to suck a certain amount of solution out of a test tube or a stock. This can only be accomplished if the cannula hand is attached and placed inside a test tube or a stock. Otherwise, although the PullSyringe does not cause robot movement collision, it generates a logical error.

Layer 2: The Standard Robot Commands. With the intrinsic functions, robot laboratory activities can be planned and controlled by a computer program. But programming using intrinsic functions is tedious and requires special attention to avoid robot movement errors. On the basis of the intrinsic functions, we have created a set of standard robot commands. Table 2 lists the robot commands and their functional descriptions for the current robotic laboratory. Programming using robot commands generates no robot collision errors and covers much of the robot activity scheduling.

A robot command may consist of several intrinsic functions. Figure 12 shows the structure of the robot command *GetHand1*. The GetHand1 command schedules the robot movements to safely attach the general purpose hand. It first checks the state of the robot hand. If *hand1* (the general purpose hand) has already been attached, the command terminates with no action. If the hand is empty, it simply attaches the general purpose hand and sets the state of the robot hand to be hand1 attached. If the *hand2* (the syringe hand) is attached, it first parks the syringe hand and then attaches the hand1. If the *hand3* (the cannula hand) is attached, it parks the cannula hand first and then attaches the general purpose hand.

A robot command can also call several other robot commands along with the layer 1 intrinsic functions. Figure 13 shows the

Table 2. Robot Intrinsic Functions

robot hands	
Attach GP	attach the general purpose hand
ParkGP	park the general purpose hand
AttachCan	attach the cannula hand
ParkCan	park the cannula hand
AttachSyr	attach the syringe hand
ParkSyr	park the syringe hand
MoveSyr(p)	move the syringe to position p
stock rack	
StockRack(index)	move the robot hand (GP/syringe/cannula) above the specified (index) stock
IntoStock( )	move the robot hand (syringe/cannula) into the stock
testtube rack	
TesttubeRack(index)	move the robot hand (GP/syringe/cannula) above the specified (index) test tube
DownToTesttube( )	move the robot hand (GP) down for grabbing a test tube
IntoTesttube( )	move the robot hand (cannula hand) into a test tube for transferring solution out
JustinTesttube( )	move the robot hand (cannula/syringe) into a test tube for transferring solution in
tip racks	
TipRack(index)	move the robot hand (syringe hand) above the specified tip
DownToTipRack( )	move the robot hand (syringe hand) down to the tip rack for griping a new tip
balance	
OpenDoor( )	move the robot hand (general purpose hand) to open the door
AboveBalance( )	move the robot hand (general purpose hand) above the balance
IntoBalance( )	move a test tube (grabbed by the general purpose hand) into the balance
ReadBalance( )	get the weight of the substance in the balance
liquid dispensing station	
PullSyringe(amount)	suck certain amount of solution from a stock or from a test tube
PushSyringe(amount)	release certain amount of the solution into a test tube
waste dispensing station	
AboveWaste( )	move the robot hand (cannula hands) above the waste dispensing station
votex station	
AboveVotex( )	move a test tube (grabbed by the general purpose hand) above the votex station
IntoVotex( )	move a test tube (grabbed by the general purpose hand) into the votex
VotexOn( )	turn the votex on
VotexOff( )	turn the votex off
centrifuge	
AboveCentrifuge(index)	move test tube (grabbed by the general purpose hand) above the centrifuge
IntoCentrifuge( )	move the centrifuge tube into the holder
CentrifugeOn( )	turn on the centrifuge
CentrifugeOff( )	turn off the centrifuge
CentrifugeSlot(number)	move the centrifuge slot
S/P water bath	
BathRack(index)	move the robot hand (GP) above the specified water bath position
DownToBathRak( )	move the robot hand down to the water bath rack
SetTemperature(temp)	set the water bath temperature
spectrophotometer	
AboveSpec( )	move the robot hand (GP) above the spectrophotometer
IntoSpec( )	move the robot hand (GP) down to the spectrophotometer
MeasureAbs(w)	measure the absorbance at the wavelength (w)
Scan(from end)	scan the absorbance at the range of from to end
pH meter	
UnderElectrode( )	move a test tube (grabbed by the GP) underneath of the electrode
IntoElectrode( )	move the test tube upward so that the electrode goes into the test tube
ReadPH( )	get the pH measurement of the pH meter
CleanElectrode( )	clean the electrode after a measurement

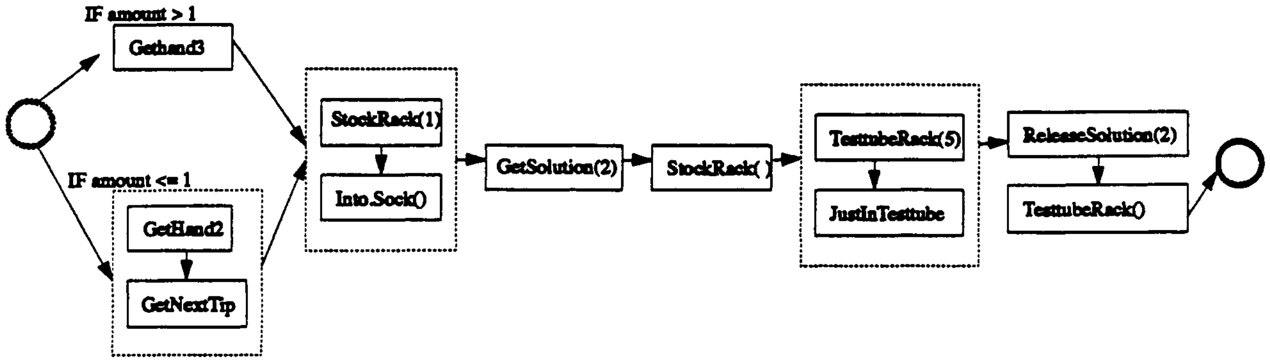


Figure 13. Structure of the robot command StockToTube.

execution path of the command *StockToTube*(*stock-1 testtube-5 2*). This command transfers 2 mL of solution from stock-1 to testtube-5. It first checks the amount of the solution being transferred. If the amount of solution being transferred is larger than 1 mL, the robot arm picks up the cannula hand. Otherwise the robot arm picks up the syringe hand and attaches the next clean syringe tip. After the proper hand (the cannula hand for this example) is attached, the robot executes the *StockRack*(*1*) intrinsic function and moves the hand above the stock-1. The robot then executes IntoStock( ) and moves

Table 3. Robot Commands

GetHand1( )	safely attach the general purpose hand
GetHand2( )	safely attach the cannula hand
GetHand3( )	safely attach the syringe hand
ParkHand( )	park the attached hand
GetWeitht(**t**)	measure the weight of test tube **t**
GetSolution(**a**)	suck certain amount of solution from a test tube or a stock
ReleaseSolution(**a**)	release certain amount of solution to test tube
StockToTube(**s t v**)	transfer **v** mL of solution from the stock **s** to the test tube **t**
TubeToTube(**f t v**)	transfer **v** mL of solution from test tube **t** to test tube **t**
TubeGoVotex(**t**)	move the test tube **t** to the votex station
Dilute(**t v m**)	dilute the test tube **t** to volume **v** by adding **m** mL of solvent
DiluteTo(**t v**)	dilute the test tube **t** to volume **v** by adding a proper amount of solvent
BackFromVotex( )	move the test tube from votex back to the test tube rack
TubeGoBath(**t b**)	move the test tube **t** from test tube rack to slot **b** of the water bath
BackFromBath( )	move the test tubes from the water bath back to the test tube rack
TubeGoCentrifuge(**t c**)	move the test tube **t** from the test tube rack to slot **c** of the centrifuge

the cannula tip into stock-1. Then the intrinsic function *GetSolution(2)* is activated to get 2 mL of solution into the cannula syringe tubing. The next intrinsic function stockrack( ) moves the syringe tip back above the stock-1. This will prevent bending of the syringe tip from the next repositioning move with the syringe tip inside the stock-1 container. The intrinsic function *TubeRack(5)* then moves the cannula hand above testtube-5. The cannula syringe is then placed inside the testtube-5, and 2 mL of solution is released with the robot command *ReleaseSolution(2)*. Finally the robot lifts up the cannula hand above the testtube-5 for safety purposes.

Robot commands are independent of the robot laboratory experiment layout. This implies that they are compatible with different equipment layout. There is no need to redefine the layer 2 robot command functions for a robot laboratory with a different equipment layout but the same hardware configuration.

Some of the Robot commands are still equipment specific. For example, there are three robot commands defined for centrifuging, *CentrifugeTwo*, *CentrifugeFour*, and *CentrifugeSix*. The CentrifugeSix command requires a centrifuge with a six position rotor.

Robot commands eliminate robotic collision errors. This is achieved by checking and updating the state of the relevant devices or instruments. For example, the GetHand1( ) command checks the state of the robot hand and parks the other hands (if they are attached) before attempting to pick up the general purpose hand. After the general purpose hand is picked up, it updates the state of the robot hand. Consider a test tube device as another example. A test tube can be in one of the six possible locations: in its slot on the test tube rack, in the test tube holder of the stirrer, in the cell holder of the spectrophotometer, in the corresponding position of the water bath, in one of the six positions of the centrifuge rotor, or being held by the general purpose hand. By updating the state of a test tube due to each operation, the robot schedules its movement to avoid collision error. The *TubeGoSpec*(*tube-4*) robot command moves the testtube-4 into the cell holder of the spectrophotometer. This robot command also updates the state of the spectrophotometer to indicate that testtube-4 is in the cell holder. Thus, a successive command *BackFromSpec* moves the testtube-4 back to slot 4 of the test tube rack.

Robot commands reduces the possible state changes of the robotic table. By preventing the robot interface module from calling the robot intrinsic functions directly, the possible state space of the robotic table is reduced. Without constraint, the syringe hand has four possible states: parked with a syringe tip, parked without a syringe tip, attached to robot arm with a syringe tip, or attached to robot arm without a syringe tip. The state of a parked syringe hand with a syringe tip is eliminated by calling *TakeOffTip* function every time *ParkSyringe* is invoked. Further, the state of an attached syringe hand without a syringe tip attached is eliminated because the *GetHand2* is always followed by the *GetNextTip* command.

Robot commands include scheduling mechanisms to avoid some of the unnecessary operations. Consider AttachNextTip( ) command as an example. The semantic of the command is defined as the following: use the old syringe tip if there is an old syringe tip attached and the old syringe tip has been used to transfer same solution being transferred; otherwise drop off the old syringe tip and pick up a new one. Refer to Figure 13; GetNextTip( ) is called after the command GetHand2. When the command GetHand2 is called, the syringe hand is either attached or parked. If the hand2 is parked, then the robot picks up the syringe hand and gets a new syringe tip. If the hand2 is attached already, then there are two cases. In the first case, the attached syringe hand has been used to transfer the same stock solution being transferred. In the second case, the syringe hand has been used to transfer a different stock solution. The AttachNextTip command uses the old syringe tip for the first case and uses a new syringe tip for the second case.

Layer 3: Conceptual Laboratory Execution. The layer 1 and 2 robot programming is implemented on the robot controller. Current commercial automation software has focused on programming the robot controller and can be used to implement layer 1 and layer 2 of the proposed protocol. The robot controller accepts a sequence of commands from the conceptual model and controls the robot execution. The layer 3 specifies how the conceptual model translates a conceptual procedure to a sequence of robot commands.

A conceptual procedure consists of a set of objects each representing a procedure step. A set of robot interference functions are defined in the conceptual model for the PG to translate a procedure step object into a sequence of robot commands. Table 3 lists the robot interface functions and their functional descriptions.

The robot interface functions at layer 3 are robot hardware configuration independent and can manage the robot execution by sending messages to the robot controller to activate the layer 2 robot commands.

The procedure generator schedules the hardware resources. Refer to the English description in Figure 6; there is only one test tube testtube-21 used in representing the conceptual procedure. To generate a robot procedure, more than one test tube are needed, depending on the number of standards and samples for a particular application. In the following discussion, we assume an application with five standards and two samples.

Table 4. Robot Interface Functions in the Conceptual Model

CentrifugeTubes(**tubes s t**)	centrifuge a list of test tubes at centrifuge speed **s** for **m** min
SeparateMixture(**t l**)	separate a precipitate at test tube **t** and transfer the liquid to test tube **l**
GetWeight(**tube**)	get the weight of the test tube **tube**
MeasureAbs(**tube w**)	measure the absorbance of the test tube **tube** at wavelength **w**
Measure_pH(**tube**)	measure the pH of the test tube **tube**
FindPeak(**tube**)	find one or more absorption peaks of the test tube **tube**
Stir(**tube s m**)	stir the test tube **tube** at stirring speed **s** for **m** min
SetTemperature(**t**)	set the water bath at temperature **t**
ControlTemperature(**tubes t m**)	control temperature of several test tubes at temperature **t** for **m** min
StockToTube(**s t a**)	transfer certain amount **a** solution from the stock **s** to the test tube **t**
TubeToTube(**f t a**)	transfer certain amount **a** solution from test tube **f** to test tube **t**
Wait(**t**)	pause for robot laboratory for **t** min before executing the next robot command
Dilute(**tube v**)	dilute the test tube **tube** to volume **v** by adding the default solvent
AbsTitrate(**tube w i lambda**)	titrate the test tube with titrant **w** by monitoring absorbance
pHTitrate(**tube w i**)	titrate the test tube with titrant **w** in increments of **i** by monitoring pH

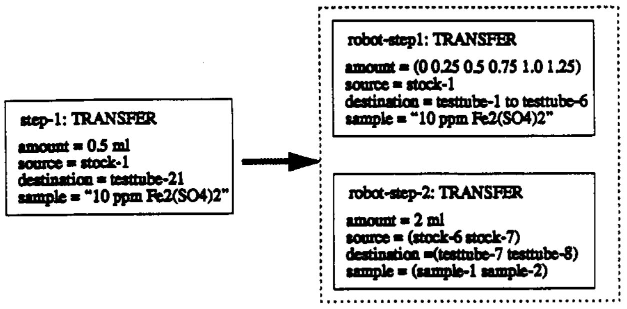

step-1: TRANSFER
amount = 0.5 ml
source = stock-1
destination = testtube-21
sample = "10 ppm Fe2(SO4)2"
robot-step1: TRANSFER
amount = (0 0.25 0.5 0.75 1.0 1.25)
source = stock-1
destination = testtube-1 to testtube-6
sample = "10 ppm Fe2(SO4)2"
robot-step-2: TRANSFER
amount = 2 ml
source = (stock-6 stock-7)
destination =(testtube-7 testtube-8)
sample = (sample-1 sample-2)

Figure 14. Generating robot procedure steps from a conceptual procedure step.

The PG generates a complete robot procedure based on the conceptual procedure and the new application requirement (five standards and two samples). A complete robot procedure consists of a set of objects, each representing a robot executable procedure step. Consider the first step of the conceptual procedure "transfer 2 mL of $Fe_2(SO_4)_3$ from stock-1 to testtube-21". The object of this step is characterized by its source being the stock-1, amount being 2 mL, and destination being testtube-21. To create objects representing the corresponding robot procedure steps, the PG interacts with the system knowledge base and concludes that five standards, two samples, and one blank need to be prepared at appropriate test tubes. Eight test tubes are then scheduled for transferring standards, samples, and blank. Two objects are created as in Figure 14. One object represents the robot operation of transferring a series of standards to testtube-1, to testtube-2, and to testtube-6 (notice that testtube-1 is used as blank). The other object represents the operation for transferring sample-1 and sample-2 to testtube-7 and testtube-8.

The robot execution of a robot procedure step is accomplished by activating a proper robot interface function. The robot-step1 object will activate the transfer interface function with three arguments. The source argument is stock-1. The amount argument is a list of numerals specifying volume in milliliters. The destination is a list of test tubes from testtube-1 to testtube-6. According to the source, the amount, and the destination arguments, the transfer interface function generates a series of StockToTube robot commands to control the robot execution. Figure 15 shows the simplified object specification, the robot interface function called, and the sequence of robot commands generated.

At the conceptual laboratory execution layer, robot programming benefits directly from the system knowledge base. Each real instrument and device has an associated object in the conceptual model. The conceptual laboratory model "knows" the state of the real device or instrument. High-level robot execution error checking and scheduling are thus possible.

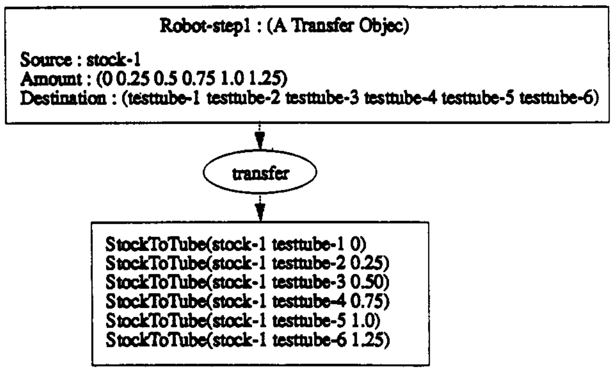

Robot-step1 : (A Transfer Objec)
Source : stock-1
Amount : (0 0.25 0.5 0.75 1.0 1.25)
Destination : (testtube-1 testtube-2 testtube-3 testtube-4 testtube-5 testtube-6)
transfer
StockToTube(stock-1 testtube-1 0)
StockToTube(stock-1 testtube-2 0.25)
StockToTube(stock-1 testtube-3 0.50)
StockToTube(stock-1 testtube-4 0.75)
StockToTube(stock-1 testtube-5 1.0)
StockToTube(stock-1 testtube-6 1.25)

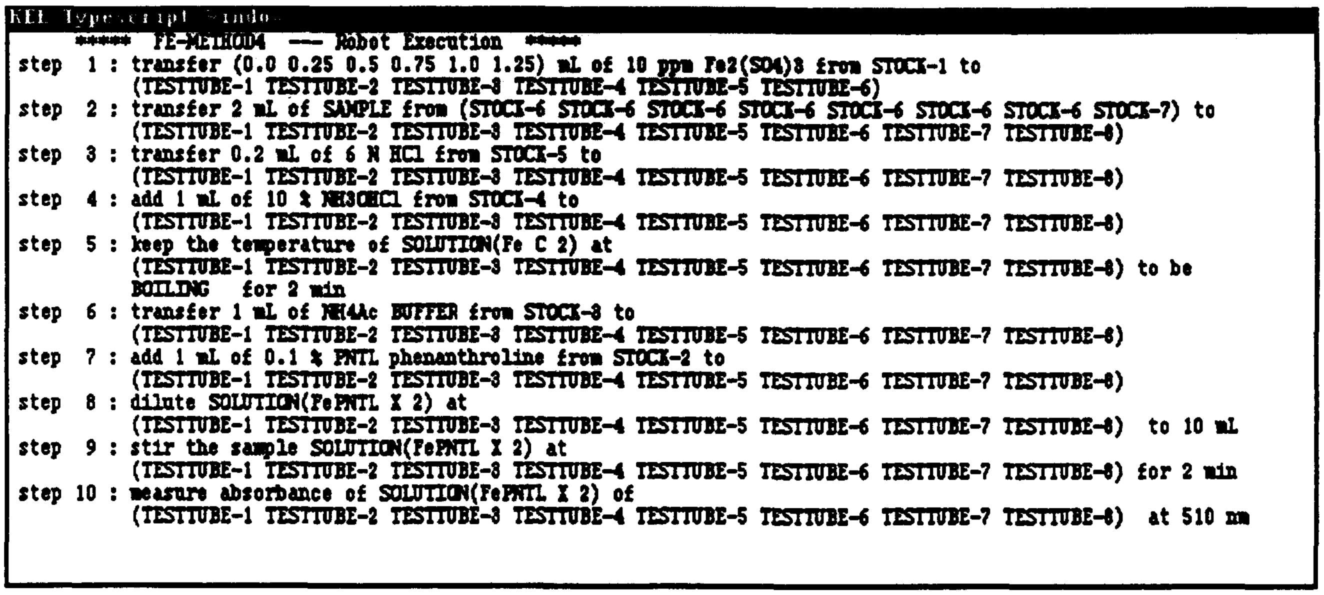
Figure 15. Translation from a robot procedure step to a sequence of robot command.

In order to centrifuge one precipitate, another test tube needs to be scheduled to achieve a balanced centrifuge operation. The PG first schedules an unused test tube. Then a certain amount of water is added into the scheduled test tube according to the volume of the precipitate–solution mixture. Finally, the robot command CentrifugeTwo is activated to separate the precipitate mixture.

The conceptual model includes much of the intelligent robot execution scheduling. And its robot scheduling mechanism can be modified by updating the system knowledge base. Examples are to perform the robot analysis with the standard addition method and to generate robot procedures with a time constraint (measuring absorbance within a time limit after the color development step). Figure 16 shows an English description of a different robot procedure of the same conceptual procedure but using standard addition method. Notice that all the standards contain samples to compensate the sample matrix effect.

Robot error checking at the conceptual laboratory execution layer is mainly based on the volume constraint, physical state constraint, and chemical state constraint. With the volume constraint, the PG prevents an operation from transferring more solution out of a test tube than it actually contains. The PG also rejects an operation of transferring certain amount of solution into a test tube and causing the total volume of the test tube exceeding the maximum. With the physical state constraint, the PG rejects an operation of transferring certain amounts of chemicals out of a test tube where the content is an uncentrifuged precipitate. With the chemical state constraint, the conceptual laboratory keeps track of the chemical components of test tubes and stocks. The PG rejects an operation of measuring absorbance of a test tube which contains no chemical component giving absorption at the specified wavelength.

Layer 4: User Interface to the Conceptual Laboratory. The fourth layer is built on the top of the third layer. A user



```
KEE Typescript Window
     *****  FE-METHOD4  --- Robot Execution  *****
step  1 : transfer (0.0 0.25 0.5 0.75 1.0 1.25) mL of 10 ppm Fe2(SO4)3 from STOCK-1 to
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6)
step  2 : transfer 2 mL of SAMPLE from (STOCK-6 STOCK-6 STOCK-6 STOCK-6 STOCK-6 STOCK-6 STOCK-6 STOCK-7) to
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8)
step  3 : transfer 0.2 mL of 6 N HCl from STOCK-5 to
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8)
step  4 : add 1 mL of 10 % NH3OHCl from STOCK-4 to
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8)
step  5 : keep the temperature of SOLUTION(Fe C 2) at
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8) to be
          BOILING   for 2 min
step  6 : transfer 1 mL of NH4Ac BUFFER from STOCK-8 to
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8)
step  7 : add 1 mL of 0.1 % PNTL phenanthroline from STOCK-2 to
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8)
step  8 : dilute SOLUTION(FePNTL X 2) at
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8)  to 10 mL
step  9 : stir the sample SOLUTION(FePNTL X 2) at
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8) for 2 min
step 10 : measure absorbance of SOLUTION(FePNTL X 2) of
          (TESTTUBE-1 TESTTUBE-2 TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6 TESTTUBE-7 TESTTUBE-8)  at 510 nm
```

Figure 16. Robot procedure generated for standard addition method.

designs a robot executable procedure at the conceptual level of using chemistry. All the lower level robot programming (layer 1 to layer 3) is hidden from the user. A conceptual procedure designed at this layer is independent of the real robot laboratory configuration. This assures the flexibility of a robot laboratory dealing with changes of instruments, robot hardare modules, setup requirements, and execution constraints. The conceptual model thus increases productivity of designing robot procedures and portability of the designed procedures. The icon-based user interface presented in a previous paper is an example designing standard robot procedures using chemistry notation and concept.[2] Using the Analytical Director expert system, different robot laboratories can read a procedure in English description as in Figure 7 and perform the robot analyses with same quality assurance.

CONCLUSIONS

The robot intrinsic functions and the standard set of robot commands extend the automation capability of a robot laboratory. The intrinsic functions define the fundamental unit operations for specific equipment and deal with the specific layout of the hardware configuration. The standard robot commands decompose the complex robot laboratory activities into a set of functional operations. A robot command accomplished its operational purpose by correlating several instruments. By imposing the time constraint of a sequenced robot movement and instrument operations, along with the state change, the standard robot commands eliminate the robot movement collision errors and some of the logical errors.

At the conceptual laboratory execution layer, the conceptual model simulates the execution of a designed procedure. This simulation results in a series of state changes of the workbench which is used by the PG to perform high-level error checking and scheduling. The PG approves a spectrophotometric procedure only if the procedure converts the target element into a chemical state which gives sensitive absorption at a given wavelength. At the conceptual laboratory execution level, the expert system can schedule the hardware resources and robot execution for a different number of standards and samples, whether or not it is using a standard addition method and other application requirements.

With the multilayer structured robot interface protocol, analytical procedures can be designed at a conceptual level using chemistry knowledge instead of robot programming. This increases both the productivity and portability. Analytical procedures represented in English language format or in object specification can be delivered to other robot laboratories and be executed. The following are some of the scenarios in applying such a robot interface protocol to laboratory automation:

(1) Same Expert System and Robot Hardware Configuration but Different Laboratory Layout. A different robot laboratory of same equipment usually implies a different layout of the robot laboratory equipment. Only the robot intrinsic functions need to be modified due to changes of those robot movement absolute positions. Following a structured robot interface protocol, knowledge-based expert systems once developed can be used by other robot laboratories with a minimum effort of modifying the robot interface control.

(2) Different Expert Systems but Same Laboratory Hardware Configuration and Layout. As long as the expert systems follow the same standard of the robot interface protocol, the same robot hardware resource can be controlled by different expert systems to perform automated analyses.

(3) Same Expert System but Different Robot Hardware Configuration. Only the standard set of robot commands and the robot intrinsic functions need to be modified to reflect the change of those particular robot control statement. Once the standard robot commands and robot intrinsic functions are refined the robotic expert system works as before. This means that robot equipment can be updated without any change of the user interface. The users will design and execute procedures in exactly the same way.

(4) Different Expert Systems and Different Hardware Configuration. This brings us to an ultimate question: Can procedures be written in a standardized format so that different robotic expert systems can read an analytical procedure and control the execution with same quality assurance? This paper has shown an example of using object specification to represent analytical procedures and has shown both the SUN- and PC-based expert systems can execute such a procedure.

The system developed has been used to convert "textbook" analytical procedures and to design new procedures. Spectrophotometric procedures for analyzing some of the common metallic element such as Mn, Cu, Fe, Ni, Mg, Ag, and Pb have been designed and saved as conceptual procedures. These procedures have been executed by the robot laboratory both using the PC-based and the SUN workstation-based expert systems. The accumulated robot analyses have demonstrated the benefit of the standard robot interface protocol: procedures designed in the conceptual level can be delivered to other robot laboratories and executed with same quality assurance.

REFERENCES AND NOTES

(1) Bleyberg, M. Z.; Zhou, T.; Isenhour, T. L.; Marshall, J. The Design and Implementation of an Analytical Chemistry Expert System. *The Third International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems*; Charleston, SC, July 1990; pp 1073–1078.

(2) Zhou, T.; Isenhour, T. L.; Zamfir-Bleyberg, M.; Marshall, J. C. Object-Oriented Programming Applied to Laboratory Automation. 1. An Icon Based User Interface for the Analytical Director. *J. Chem. Inf. Comput. Sci.* **1992**, *32*, 79–87.

(3) Zhou, T.; Isenhour, T. L.; Marshall, J. C. Object-Oriented Programming Applied to Laboratory Automation. 2. The Object-Oriented Chemical Information Manager for the Analytical Director. *J. Chem. Inf. Comput. Sci.* **1993**, *33*, 569–576.

(4) Rumbaugh, J.; Blaha, M.; Premerlani, W.; Eddy, F.; Lorensen, W. Object-Oriented Modeling and Design; Prentice Hall: Englewood Cliffs, NJ, 1991.

(5) Shank, R.; Riesback, C. R. *Inside Computer Understanding*; Lawrence Erlbaum Associates: Hillsdale, NJ, 1981.

(6) *Standard Methods for the Examination of Water and Wastewater*, 16th ed.; American Public Health Association, American Water Works Association, Water Pollution Control Federation: Washington, DC, 1985.