

A Web-based Simulation Environment for Manufacturing Education

Jeff Rickel,¹ Maged Dessouky,² Edward Kazlauskas,³

Narayanan Sadagopan,¹ Erin Shaw,¹ and W. Lewis Johnson¹

¹ *Information Sciences Institute, University of Southern California*

² *Department of Industrial and Systems Engineering, University of Southern California*

³ *Department of Instructional Technology, University of Southern California*

rickel@isi.edu, maged@usc.edu, kazlausk@rcf.usc.edu

narayans@pollux.usc.edu, shaw@isi.edu, johnson@isi.edu

Abstract. To better prepare manufacturing students for the complexity of modern factories, a Web-based Virtual Factory Teaching System (VFTS) was developed to complement traditional classroom lectures. VFTS allows students to make the decisions required to run a realistic factory and see the consequences of their decisions via an animated simulation. An automated lab instructor monitors student simulations and intervenes opportunistically with questions and explanations to ensure that students draw the appropriate connections between simulation results and underlying manufacturing principles. This paper describes VFTS and its evaluation at four universities.

1 Introduction

Traditional approaches to manufacturing education do not adequately prepare students for the complexity of modern factories. Although they learn techniques for forecasting product demand, planning production levels that can meet the anticipated demand, and scheduling the planned production at factory workcenters, each technique is typically studied in isolation, so students rarely develop an intuitive feeling for the interdependencies among these three activities. Furthermore, factory situations are typically oversimplified to allow students to make calculations by hand. Students typically learn a variety of theoretical models without acquiring an understanding of how and when they apply to complex factory situations. Without this understanding, students are likely to ignore or misuse the models as practicing engineers.

Our approach to this problem is to complement traditional classroom lectures and homework with a Web-based Virtual Factory Teaching System (VFTS). Students or their instructors can define realistic factories by specifying the properties of products and machines. Students make forecasting, planning, and scheduling decisions for their factory, and they see the consequences of those decisions via an animation window that shows the simulated factory dynamics, including products moving from machine to machine, queues rising and falling, machines becoming busy or idle, and machines going down for repair. At the end of the simulation, they can view a variety of summary statistics such as average cycle time and work in process, and they can compare these statistics to those from previous simulations to evaluate the effects of their decisions. Through this hands-on application of the theoretical models

they are learning, they develop a deeper understanding that allows them to bridge the gap between theory and practice in a realistic factory environment.

To ensure that students draw the appropriate connections between simulation results and underlying manufacturing principles, an automated lab instructor monitors their simulations and intervenes opportunistically with questions and explanations. The central pedagogical principle behind our approach is that this agent should teach domain principles in the context of the student's own decisions and simulations. To support such a tutorial style, the agent must be able to recognize learning opportunities in the student's problem-solving activities, test the student's understanding, and relate particular simulation results to general domain principles. We achieved these goals by applying our previously developed, domain-independent automated lab instructor (ALI) [4] to VFTS. Previously, ALI had been applied to chemistry and biology simulations; its application to VFTS provides further evidence of its generality.

VFTS has been used by hundreds of students in undergraduate engineering courses at four universities, as a significant portion of their course activities and grade. It has been formally evaluated at three of those schools, including its effect on student attitudes towards the class and subject, their perception of its usability and value, and both subjective and objective assessments of student learning. Both the development and evaluation of VFTS were guided by an interdisciplinary team including computer scientists, industrial engineers, and educational theorists.

In the remainder of this paper, we discuss related research (Section 2), the VFTS software architecture (Section 3), our application of ALI to VFTS (Section 4), and evaluation results from the participating universities (Section 5).

2 Related Work

VFTS is a simulation-based learning environment, and it shares its basic pedagogical approach with the many prior systems in that genre. However, it differs in a variety of details. Like simulation environments for scientific inquiry [1], our goal is for students to understand the relationships between independent variables (e.g., scheduling decisions) and dependent variables (e.g., performance measures such as cycle time and machine utilization). However, students learn these relationships through problem-solving activities (i.e., trying to improve the performance of their factory) rather than through explicit hypothesis formation and testing. In this sense, VFTS is more like systems such as the Recovery Boiler Tutor (RBT) [9], which exposes students to equipment failures and allows them to take actions and see their consequences. However, whereas RBT and similar systems present students with different scenarios and allow them to choose discrete actions to solve problems, VFTS is fundamentally a design system, in which students try to optimize various factory performance measures through their choices across a wide range of control parameters and decisions. Thus, it is closest in spirit to CyclePad [6], a learning environment where engineering students learn principles of thermodynamics by designing thermodynamic systems. However, the two systems differ in their technical details; it does not appear that the techniques used in either one would support the other, and the approaches to coaching are complementary. For manufacturing education specifically, there is a wide variety of software available for factory analysis and simulation, and some of it has been used for education (e.g., [7]), but VFTS is novel in its integration of forecasting, planning, and scheduling and its inclusion of an automated lab instructor.

3 VFTS Architecture

The VFTS software includes a variety of components, as shown in Figure 1. The VFTS client, which runs in a Web browser from the VFTS Web site (<http://vfts.isi.edu>), provides the user interface that handles input (allowing students to enter data, make decisions, manipulate views, and interact with ALI) and output (displaying forecasting and planning results, displaying the factory animation, displaying simulation results in tabular and graph form, and running ALI). The VFTS server performs actions and computations based on the student's input: it reads from and writes to the VFTS database (which includes data defining the factories, student models maintained by ALI, system log files and user account information) and interfaces to the external software packages (AweSim for simulating factories, Drasys for solving linear programs, and Ptolemy for graphing data). The VFTS Web site also provides access to course materials, project information, system documentation, and the VFTS tutorial and help information.

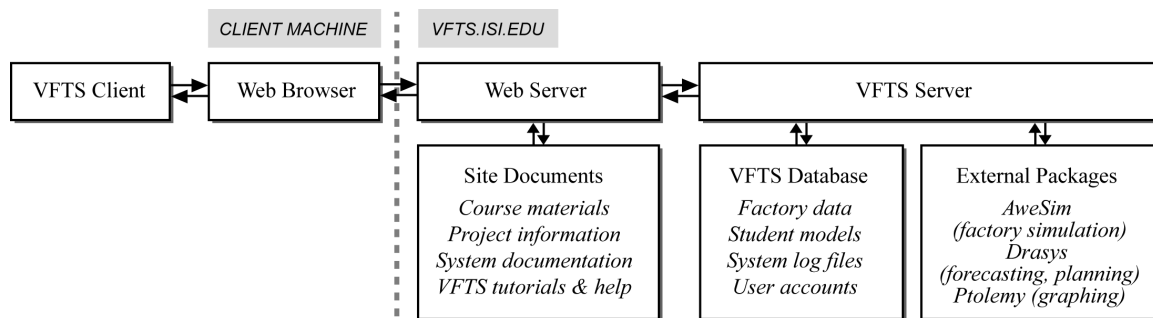


Figure 1: The VFTS architecture

Students create and run their factories via the client-side graphical user interface. To define a factory, a student or instructor specifies information such as the number of machines, the properties of each machine (e.g., average time between failure and repair time), the number of products, the properties of each product (e.g., historical demand, inventory cost, back-order cost, the sequence of machines required to produce it, and the average processing time on each of those machines), and the transportation time to move a product between each pair of machines. Students can create their own factories or edit existing factories, or instructors can create the factories and let students focus on running them.

Once a factory is defined, students use the same interface to make forecasting, planning, and scheduling decisions. They choose from among a variety of forecasting methods, and VFTS uses the historical demand and the chosen method and assumptions to forecast future demand. Next, they choose from among several planning approaches and, using the forecasted demand, VFTS formulates the planning problem as a linear program and sends it to Drasys, which outputs a production plan that VFTS displays to the student.¹ Finally, the student makes scheduling decisions, including the lot size for each product (i.e., how many to process at once on a machine before switching over to the next product), the factory release rule (i.e., the strategy for deciding when to release products onto the factory floor), and the dispatch rule for each machine (i.e., how to order products within its queue).

¹Some of the planning methods available to students do not require linear programming.

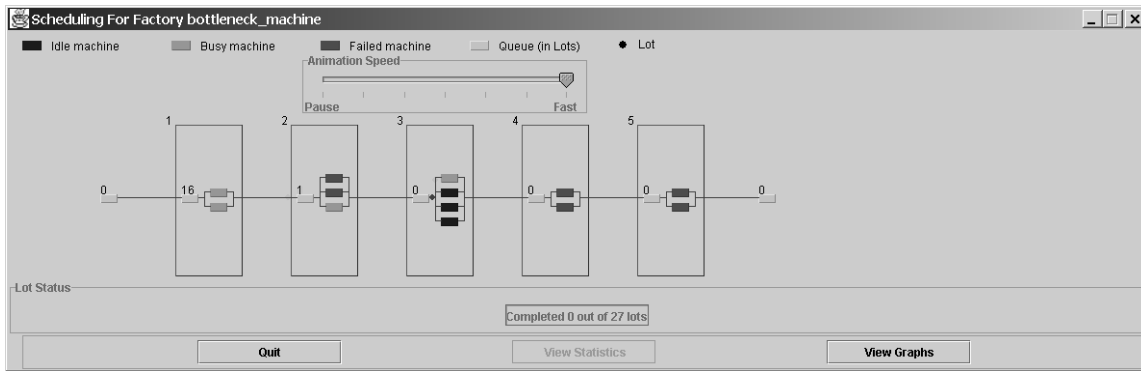


Figure 2: Simulation animation

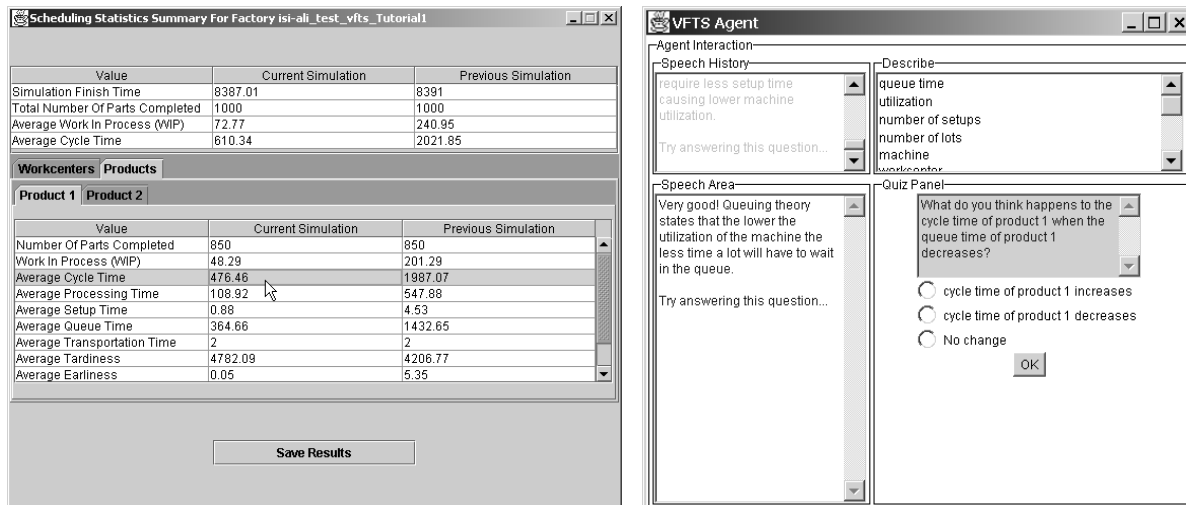


Figure 3: (Left) Simulation results

(Right) ALI's interaction window

After making these decisions, the student simulates the factory. VFTS converts the production plan (which specifies how much of each product must be made) and the scheduling decisions into the appropriate input format for AweSim, a Visual SLAM factory simulation package [8]. VFTS uses the output from AweSim to show an animation of the products moving through the factory, as shown in Figure 2. Through this animation, students see the factory dynamics, such as queues building up at bottleneck machines.

At the end of the simulation, they receive summary statistics, as shown in the left window of Figure 3. They can then iteratively change forecasting, planning, or scheduling decisions and compare the new simulation results against the previous results to see the effects of their decisions. The ability to watch the factory dynamics unfold and compare the summary statistics to previous simulations helps them develop a deeper understanding of how various decisions interact and affect factory performance. This understanding is difficult to develop with traditional lectures and oversimplified homework exercises, so it represents VFTS's greatest potential for improving manufacturing education.

Additional details on the VFTS architecture and user interface are available in an earlier paper [3]. Areas listed as future work in that paper are now fully implemented: we now support job shops (where products can move between machines in arbitrary orders) in addition to flow shops (where they move between machines in a fixed order); interprocess communi-

cation uses RMI rather than sockets for increased reliability, portability, and maintainability; forecasting, planning, and scheduling are now fully integrated (i.e., the results of one can be fed as input to the next); and we have added an automated lab instructor.

4 ALI

Ideally, students should learn how forecasting, planning, and scheduling decisions affect factory performance measures (e.g., machine utilization or average queue time) in realistic factories. In the classroom, students learn manufacturing principles and theoretical models that govern this connection between decisions and performance measures, and VFTS gives them an opportunity to connect that theory to practical situations. However, left to themselves, students may or may not draw these connections. Because students can use VFTS at any time and from any Internet computer, we cannot rely on human instructors or teaching assistants to help them make the connections. Instead, we need an automated lab instructor that is always available to watch their activities, recognize learning opportunities in their simulation results, and intervene opportunistically with questions and explanations that help them make the connections.

In prior work, we developed an automated lab instructor (ALI) [4] with precisely those capabilities. ALI is a domain-independent tutor that can be connected easily to a wide range of virtual labs. Originally, we assumed that students using such virtual labs would run simulated experiments by changing independent variables and observing the effect on dependent variables, with the explicit goal of learning the relationships among the two, as in standard scientific inquiry environments [1], and we applied ALI to chemistry and biology simulations as examples. However, although VFTS students are driven by design goals (to optimize various dependent variables by manipulating independent variables), the learning goals still revolve around understanding the relationships among those variables, so we realized that ALI's guidance would be equally applicable to learning environments like VFTS.

Applying ALI to a new domain requires two steps: (1) write a set of Java methods that interface ALI to the new simulator, so that ALI can monitor experiments, and (2) describe the simulation domain using ALI's knowledge specification language, so that ALI understands the relationships students are expected to learn [4]. The Java methods that connect ALI to VFTS were simple to provide, and our domain expert (the second author of this paper) found ALI's knowledge specification language very natural to use, so our application of ALI to VFTS went smoothly, albeit motivating several extensions to ALI discussed below.

Figure 4 shows some representative examples of ALI's knowledge. Dependent and independent variables are defined as *properties* of *entities*. High-level relationships between variables are defined as *relations*. For example, the relation in the figure specifies that increases in lot size tend to decrease cycle time, and decreases in lot size tend to increase cycle time, all other things being equal; ALI uses the qualitative functional influences (Q^+ and Q^-) of Qualitative Process Theory [5] to formalize such relationships. Finally, a high-level relation can be explained by a chain of lower-level (atomic) relations, as specified by its explanation list. For example, the relation between lot size and cycle time is explained by a chain of 5 atomic relations, one of which is shown in the figure. The textual descriptions associated with relations and atomic relations are used by ALI to explain them to students; the author can provide separate descriptions for the cases where the first variable increased or decreased (as shown in the atomic relation in the figure) or one description to use in both

```

Entity {name:product;
  description: A product refers to an item manufactured by the factory.;}
Property {name: lot size; ofEntity:product;
  description: The lot size is the size of a batch for a particular product.;}
Relation {name:lotSize cycleTime;
  properties:lot size,cycle time; type:Q-; isHeuristic:true;
  description: When the cycle time decreases as lot size increases and vice versa, it
    typically means the savings in queue time due to fewer setups dominates
    the direct effect of longer lot processing times.;
  explanationList:lotSize numberOfLots, numberOfLots numberOfSetups,
    numberOfSetups utilization,utilization queueTime, queueTime cycleTime;}
AtomicRelation {name:utilization queueTime;
  properties:utilization,queue time; type:Q+;
  increaseDescription: Queuing theory states that the higher the utilization of the machine
    the longer a lot will have to wait in the queue.;
  decreaseDescription: Queuing theory states that the lower the utilization of the machine
    the less time a lot will have to wait in the queue.;}

```

Figure 4: Examples of ALI's knowledge

cases (as shown in the first relation in the figure). Our earlier paper [4] provides more details on ALI's knowledge specification language, and a recent paper [2] describes the theoretical background behind ALI's VFTS knowledge.

ALI uses its knowledge to monitor student simulations, identify learning opportunities, and test a student's understanding, as described in detail in our earlier paper [4]. Each relation in the knowledge base represents a learning objective, and finding an example of one in the student's simulation represents a learning opportunity. When the student runs a new simulation, ALI compares the values of the independent and dependent variables to those of the previous simulation. If the student changed multiple independent variables, ALI will periodically suggest that the student change one at a time to isolate its effects. If only one was changed, ALI compares that change and all resulting changes in dependent variables to each relation in the knowledge base. If a relation and all its explanation relations are satisfied, and ALI's student model indicates that the student has not already mastered this relation, ALI initiates a dialogue. ALI first quizzes the student on the high level relation, then on each relation in its explanation chain, providing feedback in the form of general manufacturing principles, as shown in Figure 3. Finally, ALI summarizes the discussion, updating the student model based on the student's answers.

We only made a few changes to ALI's domain-independent algorithms to apply it to VFTS. First, we added the ability to mark relations as heuristic. In our earlier domains, relations represented fundamental laws, and ALI would comment on violations (e.g., due to not running a simulation to equilibrium) to prevent students from forming misconceptions. In VFTS, some relations are heuristic; if a relation is marked as heuristic, ALI will not comment on violations of it. Second, we added support for non-real-valued variables, such as a dispatch strategy, which has a discrete set of alternatives. By allowing the author to define a partial order over such alternatives, we were able to continue using Q^+ and Q^- relations in such cases (e.g., a relation might specify that choosing a dispatch strategy higher in the partial order tends to decrease cycle time). Finally, we extended ALI to recognize and discuss *instances* of relations. For example, the relations in Figure 4 apply to any product, of which there may be several; Figure 3 shows how ALI tailors its discussion to the particular product for which the relation was satisfied.

5 Evaluation

To date, VFTS has been used in senior-level industrial engineering courses at four universities: the University of Southern California (USC), San Jose State University (SJSU), the University of Virginia (UV), and Univalle University in Bolivia. The first three have been part of a controlled experiment over the past three years. During these three years, we administered a standardized set of instruments to allow us to compare across universities and conditions, including pre- and post-course attitude surveys, assessments of the student's knowledge and skills at the beginning of the course, and a final examination. During the first year, students did not use VFTS; they learned through traditional lectures and homework assignments. During the second year, they additionally used VFTS (without ALI) via a homework assignment (to become familiar with the software) and a significant project. The third year was identical to the second year except for two differences: (1) half the students at each university had ALI enabled, and (2) whereas students during the second year worked in groups within their university, students during the third year worked in inter-university teams, using groups of four consisting of two students from USC and two from SJSU (UV did not participate in the inter-university teams). Since the third-year results from UV have not been analyzed yet, we report here on our three-year results from USC and SJSU, a total of 164 students.

The difference in final exam scores between the control groups (without VFTS) and the VFTS groups was not statistically significant. This result might be expected because the exam questions were not designed to test the potential benefits of VFTS, such as a better understanding of the relationships among forecasting, planning, and scheduling. Since the final exam is a large portion of the final grade, the study team did not want to provide any questions that would put the control group at a disadvantage, so the questions were based on exams from prior years. The pre-test scores measuring student knowledge of the subject at the beginning of the course were higher (statistically significant) in the two control groups (USC and SJSU) than in the four VFTS groups, so it is possible that VFTS helped students with less prior knowledge of the subject attain equivalent final exam scores. However, we feel that a significantly revised final exam, emphasizing the deeper knowledge that VFTS is designed to provide, will be needed to clearly demonstrate objective benefits.

Instructor comments at all four universities were very positive. Through an analysis of project reports, instructors concluded that the VFTS students had a better understanding than prior students of the integration between forecasting, planning, and scheduling. The reports provided evidence that students spent a good part of their time testing many different scenarios, and students were able to see the impact of their decisions on factory performance measures. The professors also noted that a major benefit of VFTS is that it allows the students to work with a more complex factory than prior students, since the VFTS students had a more realistic project than their counterparts in the control groups. All four professors intend to continue using VFTS in their class.

Student responses on attitude surveys showed that they generally liked ALI and disliked inter-university teams. Although there was no significant difference between the ALI groups and non-ALI groups on final exam scores, the attitude surveys suggest that many students believe they benefited from ALI. They were asked to rate the following three statements on a six-point Likert scale: (1) ALI's questions and comments caused me to think about the simulation results more than I would have otherwise; (2) ALI's explanations helped me understand the material; and (3) Having ALI require me to answer his questions forced me to think about the question more than I would have otherwise. On a scale of 1 to 6 (strongly

disagree, disagree, somewhat disagree, somewhat agree, agree, and strongly agree), the mean response for these three statements at USC/SJSU was 4.08/3.5, 4.08/3.67, and 4.46/3.33, with individual responses ranging all the way from 1 to 6. When asked to comment on the experience of inter-university teams, many students liked the idea, but many commented that collaboration was difficult due to limited communication.

We also used VFTS with ALI in a workshop at the 2002 Regional IIE Student Conference, which also included industry speakers, student presentations, and factory site tours. In the VFTS workshop, students competed in a contest using VFTS to determine the best production strategies for a given factory situation. Of the 14 students completing an overall evaluation of the conference, 11 reported the VFTS workshop as the most enjoyable activity, and they reported that the software was easy to use, fun, and stimulating.

6 Conclusion

VFTS integrates several powerful pedagogical tools – simulation, an automated lab instructor, and Web-based delivery – into a novel learning environment for manufacturing education. It has been tested and improved through use by hundreds of students at four universities, where professors and students alike see it as a valuable addition to traditional lectures and homework assignments. As such, it represents a successful transition of AI-ED research into the classroom and provides a valuable test bed for continued research.

7 Acknowledgments

This research was partially supported by the National Science Foundation under grants CDA-9616373 and EEC-9872488 and by the Powell Foundation. We thank the many people that have contributed to the project, especially Sadashiv Adiga, Diane Bailey, George Bekey, Edwin Boyd, Mohammad Reza Kolahdouzan, Chat Srivaree-ratana, and Sushil Verma.

References

- [1] T. de Jong and W.R. van Joolingen. Scientific discovery learning with computer simulations of conceptual domains. *Review of Educational Research*, 68(2):179–201, 1998.
- [2] M.M. Dessouky, J. Rickel, and N. Sadagopan. An agent-based learning approach for teaching the relationship between lot size and cycle time. *INFORMS Transactions on Education*, 3(1), September 2002.
- [3] M.M. Dessouky, S. Verma, D.E. Bailey, and J. Rickel. A methodology for developing a web-based factory simulator for manufacturing education. *IIE Transactions*, 33(3):167–180, March 2001.
- [4] A. D’Souza, J. Rickel, B. Herreros, and W.L. Johnson. An automated lab instructor for simulated science experiments. In *Proc. 10th International Conference on AI in Education*, pages 65–76. IOS Press, 2001.
- [5] K.D. Forbus. Qualitative process theory. *Artificial Intelligence*, 24:85–168, 1984.
- [6] K.D. Forbus, P.B. Whalley, J.O. Everett, L. Ureel, M. Brokowski, J. Baher, and S.E. Kuehne. Cyclepad: An articulate virtual laboratory for engineering thermodynamics. *Artificial Intelligence*, 114:297–347, 1999.
- [7] U.M. Brens Garcia, D.A. Badner, et al. The virtual industrial system: A tool for learning production planning concepts and techniques. In *Proc. ASEE Annual Conference and Exposition*, 2002.
- [8] A.A.B. Pritsker and J.J. O’Reilly. *Simulation with Visual SLAM and AweSim*. John Wiley and Sons, 1999.
- [9] B. Woolf, D. Blegan, J.H. Jansen, and A. Verloop. Teaching a complex industrial process. In *Proc. 5th National Conference on AI (AAAI-86)*, pages 722–728, Los Altos, CA, 1986. Morgan Kaufmann.