Modeling Automation With Cognitive Work Analysis to Support Human-Automation Coordination

Yeti Li and **Catherine M. Burns**, University of Waterloo

Cognitive work analysis is useful to develop displays for complex situations, but it has not been well explored in providing support for human-automation coordination. To fill this gap, we propose a degree of automation (DOA) layering approach, demonstrated by modeling an automated financial trading domain, with a goal of supporting interface design in this domain. The abstraction hierarchy and the decision ladder each adopted an additional layer, mapping functions allocated to the trader and to the automation. In addition to the mapping, we marked the four stages of automation on the decision ladder to provide guidance on representing the function allocation at the task level. Next, we compared the DOA layering approach to how automation was represented in the cognitive work analysis literature. We found that a DOA-layered decision ladder, which included well-developed knowledge of the stages and levels of automation, can be suited to modern automated systems with different DOAs. This study suggests that the DOA layering approach has important implications for designing automation displays and deciding stages and levels of automation and may be a useful approach for modeling adaptive automation.

Keywords: cognitive work analysis, human-automation interaction, degree of automation, stages and levels of automation, automated financial trading, abstraction hierarchy, decision ladder

INTRODUCTION

Automated systems are becoming more pervasive, and the degree of automation (DOA) that is possible has been increasing. Recently, there has been a growing interest in artificially *intelligent automation* (Sheridan, 2017). The IBM Watson computer that defeated a human chess champion and self-driving cars are two examples suggested by Sheridan (2017). These two examples are highly automated systems powered by sophisticated machine learning algorithms (Sheridan, 2017). At a high DOA (i.e., later stages and higher levels within stages; Wickens, Li, Santamaria, Sebok, & Sarter, 2010), task dynamics are represented by automated features, but humans still take a supervisory role by initiating parameter changes that drive the control of the system. For example, financial trading algorithm designers specify the goals, knowledge, and criteria to the development process. After the automated systems are implemented, users must sometimes regain full control when unexpected automation failures or environmental disturbances occur (e.g., the driver must manually reduce speed when a self-driving car enters a road construction zone that is undocumented in the navigation system). Therefore, for highly automated systems with all stages and levels of automation (Parasuraman, Sheridan, & Wickens, 2000), there is clearly a need for occasional human intervention (Sheridan, 2017). Understanding where and how humans will interact with automation remains a challenge.

Two approaches for keeping humans in the loop have been to manipulate the DOA to either avoid high DOA situations or provide adaptive automation when users are in varying DOA contexts. These approaches are derived from the *automation trade-off* (e.g., Bainbridge, 1983; Sarter & Woods, 1995). There might be a third approach. A recent meta-analysis suggested that effective, or "ecological," displays may modify or even reverse the automation trade-off, which means that a higher DOA could improve routine and failure performances (Onnasch, Wickens, Li, & Manzey, 2014). Though preliminary, this suggestion has

Address correspondence to Catherine M. Burns, Department of Systems Design Engineering, University of Waterloo, 200 University Ave. W, Waterloo, ON N2L 3G1, Canada, c4burns@uwaterloo.ca.

Journal of Cognitive Engineering and Decision Making
2017, Volume 11, Number 4, December 2017, pp. 299–322
DOI: 10.1177/1555343417709669
Copyright © 2017, Human Factors and Ergonomics Society.

shown that designing better displays for automated systems can be a useful approach. We argue, though, that to design effective displays or to choose an appropriate DOA, one should first develop models of the cognitive work that the user will experience in different DOA contexts. These models can begin to show the functions that the user must take over in cases where the automation must be ended. Furthermore, these models could be used to help to derive the design requirements for displays that can help users work with higher levels of automation, without losing situation awareness.

Consistent with this idea, Kaber, Riley, Tan, and Endsley (2001) suggested that automated system displays must highlight the transition between system states and inform operators of the allocation of control responsibilities. This suggestion aligns with the goal of the ecological interface design approach, of making control opportunities visible to retain skill and awareness (e.g., Borst, Flach, & Ellerbroek, 2015; Furukawa & Parasuraman, 2003; Kaber et al., 2001). Borst et al. (2015) recently advanced the understanding of ecological interface design applied to automated systems, suggesting that ecological displays should coordinate with the increasing DOA by providing more information to support human-automation coordination. However, stronger approaches are needed to help determine what that information should be.

In this paper, we propose an approach to transform knowledge from the "stages and levels of automation" model to design requirements that could promote human-automation coordination. By integrating the "stages and levels of automation" model into an analysis, we can discover important properties of human-automation interaction that could be represented in better designs. Since cognitive work analysis (CWA) has shown success in determining requirements for complex systems, it makes sense to explore how CWA could be used more effectively to generate design requirements for automated systems. In this paper, we demonstrate our approach in an automated financial trading domain. Financial systems present a fertile domain to explore human decision making, with complex dynamics and increasingly pervasive automation. There have been some but not many human factors studies on financial systems in general (e.g., behavior and performance modeling: Achonu & Jamieson, 2003; McAndrew & Gore, 2013; systemic safety: Sundström & Hollnagel, 2011; incident analysis: Leaver & Reader, 2016), but none specifically analyzed automated trading. Studying automated trading presents many potential research opportunities. First, applying CWA to automated trading expands the application of CWA to a complex market-based domain that operates on different principles from physical systems (e.g., process control). Second, studying automated trading presents an opportunity to address the automation trade-off in a financial domain. In financial markets, the majority number of transactions are now completed with automation technologies, mostly through sophisticated trading algorithms (Iati, 2009). While trading algorithms improve the human ability to utilize small profitable opportunities (e.g., a small trading window may only last seconds or milliseconds), traders may encounter attentional failures while interacting with the trading algorithms or intentionally abuse them. An example of attentional failures in financial trading is a slip or lapse (e.g., Leaver & Reader, 2016), and an example of abuse of automation is "spoofing"—illegally profiting from market manipulation by generating fake supply or demand (e.g., *N. D. Ill. v. Sarao*, 2015). The last research opportunity lies in the great flexibility in developing trading algorithms. Traders may develop trading algorithms using all stages and levels of automation. Here we give two examples of trading algorithms with different DOAs. The first example is high-frequency trading, with a rigid execution algorithm to trade in milliseconds. This algorithm typically has a high DOA that requires minimal human intervention; therefore, it introduces a new risk of magnifying market value losses. For these reasons, high-frequency trading systems have received increased regulatory pressure; for instance, traders who utilize high-frequency technologies are being closely monitored by the regulators (Fabozzi, Focardi, & Jonas, 2011). In certain cases, traders may be more inclined to move toward developing intermediate DOA algorithms or manual trading (Li, Burns, & Hu, 2015). As another example, it has been reported in the literature that traders using more advanced algorithms may completely outperform and profit from their competitors who are equipped with less advanced technologies. This is a phenomenon known as "quality arbitrage" (Davis, Kumiega, & Van Vliet, 2013). We summarize the research opportunities discussed so far: The complexities of automated financial trading suggest that human factors research in this area could contribute to new understandings of human decision making and improvements to financial trading software. An investigation on how traders interact with different DOA algorithms could improve the understanding of automated financial trading and automation in general.

The remaining part of this paper is organized as follows. We first introduce two automated financial trading scenarios: one for low DOA, the other for high DOA. After that, we propose a *DOA layering approach*, showing how CWA can be used to model these scenarios by extending the work domain analysis (WDA) and control task analysis (ConTA) to account for DOA. As part of this work, we discuss the difference in the proposed DOA layering approach and how automation was represented in the CWA literature. We then discuss the implications of using the DOA layering approach—implications for display design and implications for automation design. Finally, we discuss that a future application for the DOA layering approach would be to represent function allocation that changes during DOA shifts, for modeling adaptive automation.

AUTOMATED FINANCIAL TRADING SCENARIOS

Two financial trading scenarios are used for this analysis—basket trading and trend following trading. The two trading systems differ in their DOAs and were mainly inspired by the knowledge obtained from a literature review (e.g., Chan, 2009) and a discussion with subject-domain experts.

Low DOA Scenario: Basket Trading

Basket trading systems are popular in the institutional trader community. To use a basket trading system, the trader first configures a "data analysis and order generation" algorithm to create a shortlist of financial products for trade. The trader then executes the algorithm to generate a basket of orders. On the completion of all orders in the basket, the trader may adjust his or her portfolio holdings without altering the portfolio allocation. As part of the purpose of basket trading, the basket of orders should be executed simultaneously, though price movements of the financial products are quick. The basket of orders must go through a trading platform to reach the market exchange. Either the trading platform is provided by the trader's brokerage firms (e.g., Interactive Brokers), or it is broker-neutral software (e.g., Bloomberg Terminal). Chan (2009) described the basket trading system as typically running "only a few times a day in order to generate one or a few waves of orders." This description showed that the basket trading system is a low DOA semiautomated system. The asynchronous nature of basket trading (e.g., collecting data and generating orders) is related to information analysis and decision making. In normal conditions, the order execution is synchronous with the financial market. In other cases, if it is not possible to execute all orders synchronously (e.g., in a volatile market), the basket trading system could fail. The trader may also make a wrong decision on the proportions of the financial product in the basket.

High DOA Scenario: Trend Following Trading

Trend following trading systems are real-time trading systems, typically based on a sophisticated technical analysis (e.g., moving average: Ellis & Parbery, 2005; Bollinger Bands: Bollinger, 2001). Our automated financial trading experts described a hypothetical trend-following system: A trading system uses a "scalping" algorithm based on a moving average technical analysis, seeking to make profitable trades based on arbitrage of small price gaps. The algorithm typically goes through a number of trade iterations. Once a trade iteration is completed, another iteration will begin automatically, limited only by a total number of iterations defined by the trader. The algorithm has distinct buy-and-sell logic. For example, the algorithm would wait to confirm a buying signal that the 50-day simple moving average (SMA) crosses above the 200-day SMA on the day candles and that the relative strength index in an oversold territory is <30. Once a buying signal is identified, the algorithm would place multiple buying orders in 10 iterations to the market, buying a random quantity between 400 and 800 shares in each iteration. To use a scalping algorithm, the trading platform must perform real-time data collection, automatic decision making, and rapid order placing. The scalping algorithm is perfect for exploiting a small market opportunity repeatedly without manually relaunching the trading system. The trader typically evaluates the performance of the trend-following system using a set of measures, such as Sharpe ratio, total profit or loss, and commissions. The trader has authority over monitoring every trade made by the scalping algorithm, but the monitoring is not required. The trader would typically intervene when the trading system achieves expected revenue or when the scalping algorithm needs a performance upgrade. However, the trader may override the autonomous operation, if an algorithm bug or market disturbance occurs or by canceling or modifying an order, closing a position (e.g., selling off), or stopping the entire trading system. According to Chan's (2009) description, the trend-following system has a high DOA.

USING THE WDA TO MODEL AUTOMATION

We model these two automated financial trading scenarios using WDA. We first build a base abstraction hierarchy (AH) from the domain and then propose a DOA layering approach for representing the DOA.

Base AH

We propose that a base AH be developed as is typically done in CWA. The base AH should include the usual five levels of abstraction, as in the original AH approach of Rasmussen (1986) and Vicente (1999). The scope of the base AH is limited to the system under control by the user or the automation and does not include the automation. Once developed, the base AH can serve as a template for mapping the influence of automation on the domain.

We developed a base AH to represent the financial trading domain, using the two automated financial trading scenarios. Since the descriptions of the scenarios are generally task specific, we reviewed the scenarios with our subject-domain experts and distilled the scenarios into domain functions (e.g., the functions of buying and selling in both scenarios). Later, the domain functions were organized to fit the five levels of abstraction, excluding DOA-specific functions (e.g., the basket of orders in the low DOA scenario and the moving average technical analysis in the high DOA scenario).

As a result, the base AH shows that the flow of securities is largely about buying and selling and is governed by principles such as the law of supply and demand and the flow of capital. In the next paragraphs, the base AH is described in detail, with numbers corresponding to labels in Figure 1.

1. Functional purpose. Functional purpose shows the purposes of trading. Financial activities have a commonly accepted goal, which is to make a profit. At the same time, financial activities receive regulatory constraints, such as market principles and laws. The regulatory constraints shall ensure that traders and automation are seeking to profit in legal ways.

2. Abstract function. Abstract function defines principles, priorities, and values to follow in achieving the functional purpose. We identified two groups of abstract functions: financial decision-making principles and market constraints. Financial decision-making principles include the law of supply and demand, a law governing financial activities at the most fundamental level, as named in Adam Smith's 1776 book *The Wealth of Nations*. We identified a priority that, to balance gains and losses, acknowledging that the ideal balance point of gains and losses would interact with the profit goal as well as the acceptable risk level. For example, a trader may aim to maintain a diversified portfolio to protect oneself against the risk of volatility, while other traders may seek higher profits at greater risk. Financial products must be traded ethically according to the values of the trading system. Otherwise, there could be ethical problems, such as the market crash

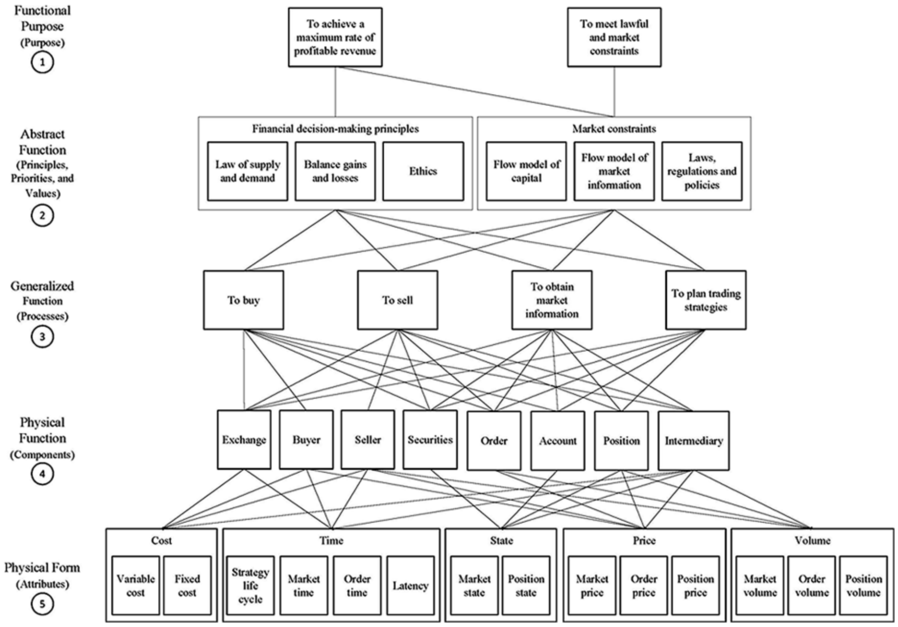
Functional Purpose (Purpose) 1
To achieve a maximum rate of profitable revenue
To meet lawful and market constraints
Abstract Function (Principles, Priorities, and Values) 2
Financial decision-making principles
Law of supply and demand
Balance gains and losses
Ethics
Market constraints
Flow model of capital
Flow model of market information
Laws, regulations and policies
Generalized Function (Processes) 3
To buy
To sell
To obtain market information
To plan trading strategies
Physical Function (Components) 4
Exchange
Buyer
Seller
Securities
Order
Account
Position
Intermediary
Physical Form (Attributes) 5
Cost
Variable cost
Fixed cost
Time
Strategy life cycle
Market time
Order time
Latency
State
Market state
Position state
Price
Market price
Order price
Position price
Volume
Market volume
Order volume
Position volume

Figure 1. Base abstraction hierarchy of financial trading.

(Davis et al., 2013). In the second set of abstract functions, market constraints, we have represented the flow of capital, market information, and laws and regulation. The flow of capital influences trading in that no trader can trade beyond one's authorized capital limit, and capital must flow among market participants to keep the market liquid. Market information must also flow to enable decisions, following a certain protocol. For example, introduced in 1992, the Financial Information eXchange protocol is the de facto electronic communication standard protocol. The Financial Information eXchange protocol regulates the information flow in a financial market, exchanging real-time trading data related to securities, orders, and trades between traders and brokers (Hu & Watt, 2014). Furthermore, the markets are subject to regulations and policies that may influence individual trades, securities, and market behavior as a whole.

3. Generalized function. At the generalized function level, we identified four main processes: (1) to buy, resulting in position gains of a portfolio; (2) to sell, resulting in position losses of a portfolio; (3) to obtain market information such as quotes and order books; and (4) to develop successful trading strategies.

4. Physical function. The physical function level shows physical components, including (1) exchange, a computerized auction market (e.g., New York Stock Exchange; traders and automation may have access to multiple exchanges, allowing them to execute arbitrage strategies across exchanges); (2) buyer and (3) seller (who can be traders or automation representing a trade client); (4) securities, identifying which financial products are being traded (multiasset trading platforms use multiple securities at the same time); (5) order, showing instructions of a trading action (a bid order represents increasing a position; an ask order decreases a position); (6) account and (7) position, showing a trader's capacity in the form of cash and assets; and (8) intermediaries, which are normally brokers offering services to a number of trade clients and market exchanges.

5. Physical form. The bottom level, physical form, shows the operational conditions, or attributes. There are five categories: (1) cost, including variable and fixed costs to trade; (2) time, showing the life cycle of a trading strategy, market and order time, and latency; (3) state of the market and the position; (4) price, including market price, order price, and price of portfolio in a certain currency; and (5) volume, including market, order, and position volume in shares. Many of these attributes can be seen directly through the trading platform visualizations.

DOA Layering on the Base AH

We have modeled the financial trading domain as broad as possible, thereby representing both automated financial scenarios with the same base AH. A consistent base AH can be used as the common ground for portraying DOA-specific information that was excluded from the base AH.

Having completed this model, we propose the DOA layering approach, layering automation on the base AH. The key to this approach is to identify the responsibility of each function in the base AH. A function in the base AH can be represented as either a sole responsibility of a trader or automation or a shared responsibility. For simplicity, in the following examples of DOA layering, we represent functions that are solely allocated to the trader or the automation, excluding shared function allocations.

The function allocation was based on domain knowledge, with much of the knowledge coming from the literature review (e.g., Chan, 2009), ethnographic experience at a trading software company, and discussions with traders on staff at the company. The first author had been involved in an observational study at Quantica Trading Inc., an automated trading software company based in Kitchener, Canada. He was part of a multidisciplinary team, including staff traders, to redesign an automated trading platform. Details of this observational study were reported in a previous paper (Li et al., 2015). In certain cases, details that would be instrumental in determining the function allocation were not available in the literature. Particularly in this domain, details about a trading system are rarely publicized, as the finance industry is unique for its strict confidentiality and protection of institutional clients. This unique characteristic of the finance industry also led to a significant limitation in being able to directly observe professional traders. To mitigate these concerns, we discussed with subject matter experts, staff traders, available at the company about function allocations that were missing in the two scenarios. For example, the high DOA scenario suggested that the functional purpose “to achieve a maximum rate of profitable revenue” would be allocated to the automation. The functional purpose “to meet lawful and market constraints” was not described in the scenario literature, but discussions with the subject matter experts suggested that this function was best allocated to the trader.

As shown in Figures 2 and 3, functions of the base AH were assigned shades. Functions allocated to the automation were shaded, and functions allocated to the trader were not shaded.

Figure 2 shows the low DOA function allocation. We can see that the higher levels—namely, the functional purpose and the abstract function—are solely allocated to the trader. The trader is responsible for deciding the proportion of each financial product in the portfolio allocation. The lower levels—generalized function, physical function, and physical form—are allocated to both the trader and the automation. The automation is not capable of controlling all aspects of trading, thereby requiring trader involvement.

In Figure 3, we present the high DOA function allocation. While the automation continues to share functions at the lower levels with the trader, it also plays a role in controlling functions at the higher levels. For example, the scalping algorithm is responsible for ensuring the profitability of the trading system (functional purpose). The algorithm may realistically achieve this purpose by balancing gains and losses (abstract function), even as the trader exercises authority over other abstract functions (e.g., “ethics” and “laws, regulations, and policies”). To manage both generalized functions of buying and selling, the algorithm must accurately choose entry and exit points into the market. A broader base of information is being considered by the algorithm, such as the market price, the order price, and the latency of the order (physical functions and physical forms).

The DOA layer adds a new dimension to the base AH, showing how human and automation work collaboratively at a certain DOA. Functions

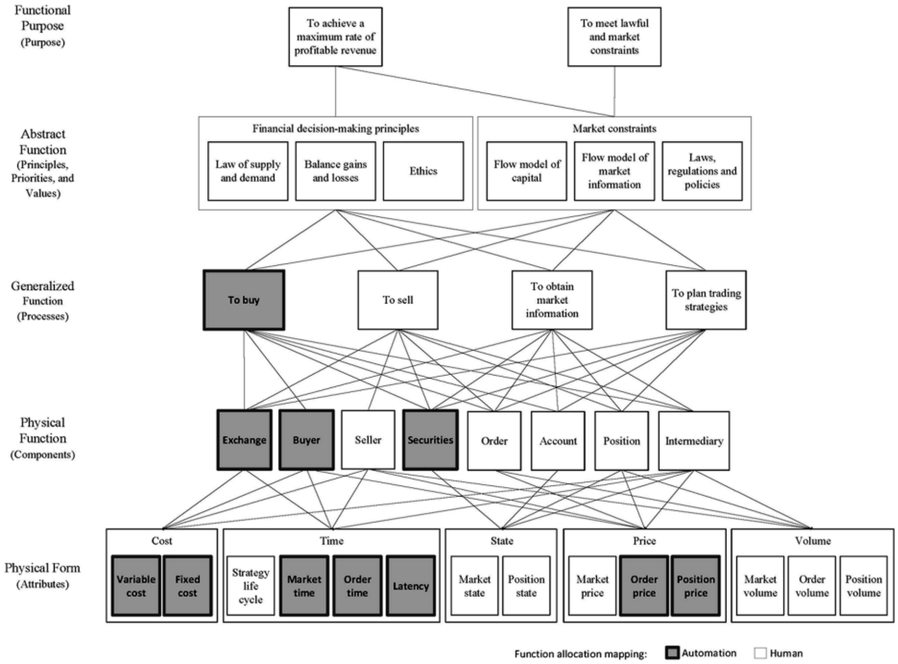
Functional Purpose (Purpose)
To achieve a maximum rate of profitable revenue
To meet lawful and market constraints
Abstract Function (Principles, Priorities, and Values)
Financial decision-making principles
Law of supply and demand
Balance gains and losses
Ethics
Market constraints
Flow model of capital
Flow model of market information
Laws, regulations and policies
Generalized Function (Processes)
To buy
To sell
To obtain market information
To plan trading strategies
Physical Function (Components)
Exchange
Buyer
Seller
Securities
Order
Account
Position
Intermediary
Physical Form (Attributes)
Cost
Variable cost
Fixed cost
Time
Strategy life cycle
Market time
Order time
Latency
State
Market state
Position state
Price
Market price
Order price
Position price
Volume
Market volume
Order volume
Position volume
Function allocation mapping:
■ Automation
□ Human

Figure 2. Abstraction hierarchy of basket trading (low degree of automation).

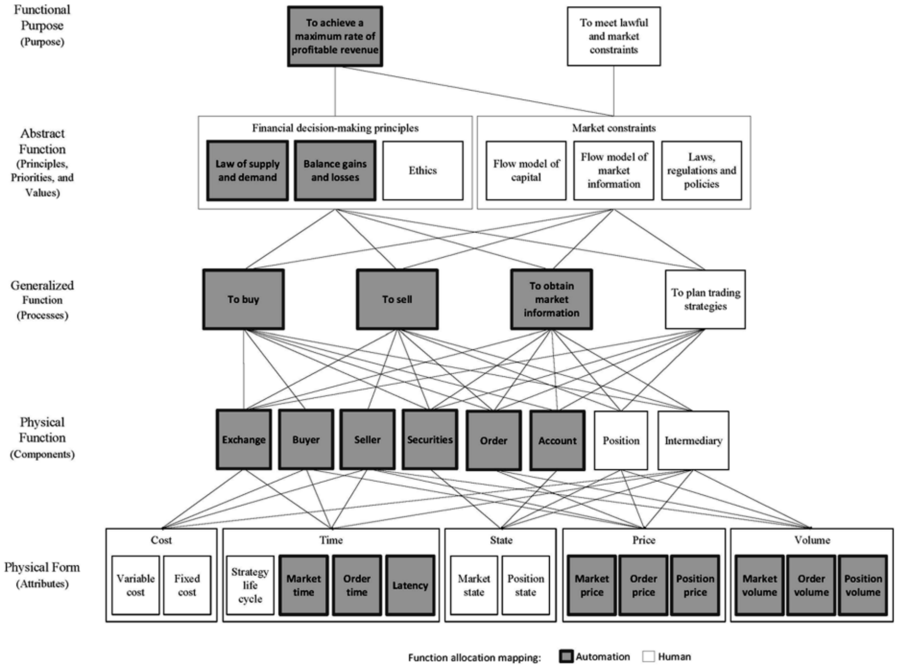
Functional Purpose (Purpose)
To achieve a maximum rate of profitable revenue
To meet lawful and market constraints
Abstract Function (Principles, Priorities, and Values)
Financial decision-making principles
Law of supply and demand
Balance gains and losses
Ethics
Market constraints
Flow model of capital
Flow model of market information
Laws, regulations and policies
Generalized Function (Processes)
To buy
To sell
To obtain market information
To plan trading strategies
Physical Function (Components)
Exchange
Buyer
Seller
Securities
Order
Account
Position
Intermediary
Physical Form (Attributes)
Cost
Variable cost
Fixed cost
Time
Strategy life cycle
Market time
Order time
Latency
State
Market state
Position state
Price
Market price
Order price
Position price
Volume
Market volume
Order volume
Position volume
Function allocation mapping:
■ Automation
□ Human

Figure 3. Abstraction hierarchy of trend following trading (high degree of automation).

can be allocated to any actor of the work domain (human or automation). Shared allocations could also be included in the DOA layer, though the analyst may want to differentiate between shared allocation approaches. The greater breadth of physical functions and associated attributes of the DOA layer can show where situation awareness losses might occur. Effective salient display of this information and the operation of the functions being performed by the automation may inform more effective displays in this situation.

USING THE CONTA TO MODEL AUTOMATION

The WDA, as described in the previous section, can map the function allocation of automation on domain structure, while the other phases of CWA can illustrate the behavior of the automation. In particular, the ConTA looks at how information is processed, mapping those task stages on to the decision ladder (DL) and exploring various shortcuts that are possible in processing (McIlroy & Stanton, 2015; Vicente, 1999). In this section, we propose utilizing the "stages and levels of automation" information while conducting a ConTA. We first discuss how to represent the four stages of automation on a base DL. This base DL is a template having the usual ladder structure and shortcuts as in an original DL. We then model four cases using a layering approach on the base DL. The four cases include two automated financial trading scenarios (the low DOA and high DOA), each in two situations (routine operations and unanticipated situations).

Representing Four Stages of Automation on the Base DL

We divided the DL into four regions and mapped the four stages of automation on to the ladder. Automation of four stages includes acquisition automation, analysis automation, decision automation, and action automation (Parasuraman et al., 2000).

We describe DL steps using Rasmussen's terminology (1974) and their affiliation with the four stages of automation, with list numbers corresponding to labels in Figure 4. The following points explain the justification for this mapping and connect DL and stages of automation contents to financial trading examples. We correlate the DL steps to the five levels of abstraction of the base AH that we previously presented.

1. Activation. The DL may start when traders are notified by environmental signals in the market (physical form: "market price"). If this DL step is automated, acquisition automation receives real-time quotes from the market (physical form: "market price") when the market is open and a reliable data connection is established.

2. Observe. Traders observe alerts from the previous step (generalized function: "to plan trading strategies") and reduce noise to form a set of observations (physical functions: "exchange," "securities," "account," "position," and "intermediary"), based on a subconscious mental model. If this step involves acquisition automation, it will become an automated data processing step based on a predefined rule. For example, an algorithm prioritizes stocks depending on their volatility (generalized function: "to plan trading strategies"), then presents the priority list to the traders for further research.

3. Identify. At this step, traders identify the underlying state of the trading system (abstract function: "flow model of capital"). For example, traders may correlate the current state to a previously experienced state. In aviation and process control domains, trend displays are provided in analysis automation to help the operators make sense of the available information (Parasuraman et al., 2000). In financial trading, similar tools (e.g., trend line and moving average) are used to help traders identify market movements (abstract function: "flow model of market information").

4. Interpret and evaluate. Rasmussen (1974) pointed out that human decision making is a "very complex mental process that requires a high level of abstraction of the domain knowledge," and expert operators may bypass this process if the system state is known. For example, a professional trader may find an association from the current state to a certain chart pattern that leads to a trading opportunity (abstract function: "flow model of capital"). A novice or nontrader does not have an ability to bypass the interpretation and must actively look for possible options (two functional purposes: "to achieve a maximum rate of profitable revenue" and "to meet lawful and market constraints"). Similarly, decision automation is related to varying numbers of options to choose from, depending on the level of automation

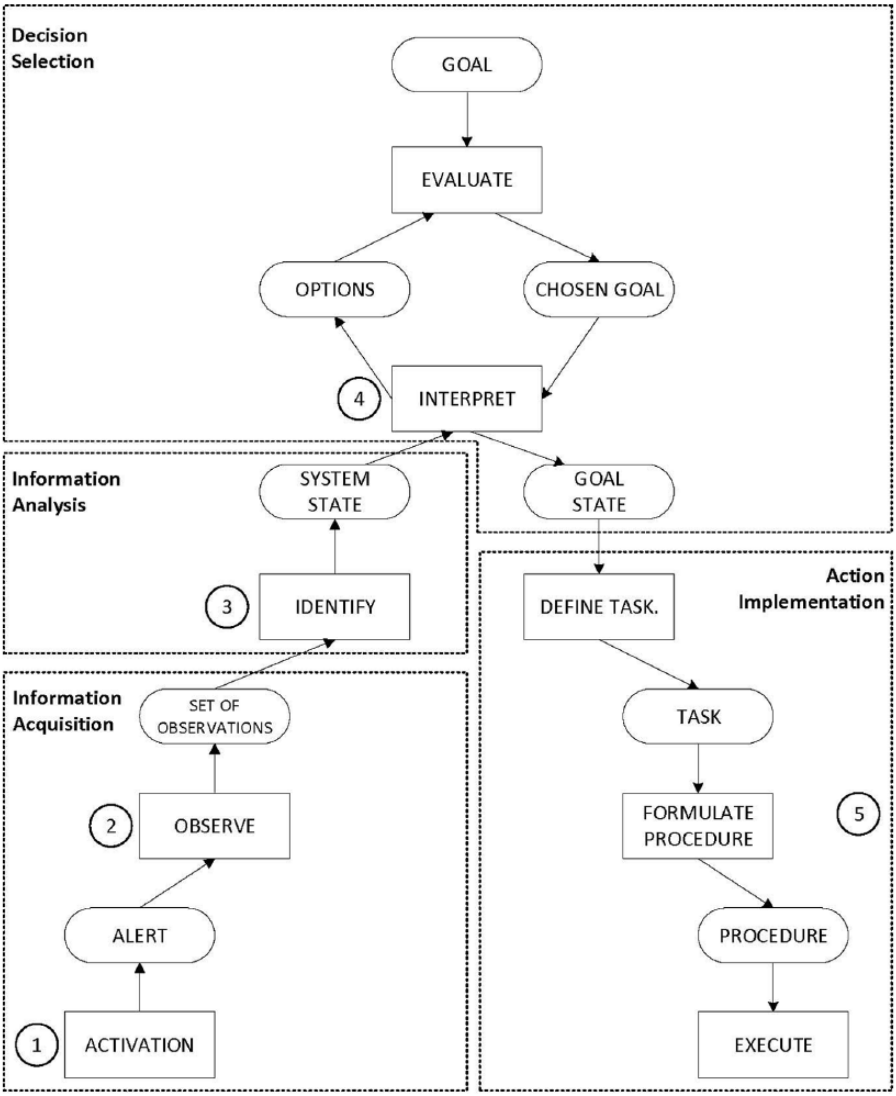

Figure 4. Representing stages of automation on a decision ladder.

(Parasuraman et al., 2000). For example, an ad hoc algorithm trades when a market indicator (e.g., price) meets certain criteria (functional purpose: "to achieve a maximum rate of profitable revenue"). The algorithm uses "If *x*, then *y*, else *z*" conditional logic; *y* and *z* are known states that can be mapped to decisions *a* and *b* separately. If the state is unknown (functional purpose: "to meet lawful and market constraints"), there will be no decision. In another example, a machine learning algorithm uses a higher level of decision automation and could be more artificially intelligent than the ad hoc algorithm. The machine learning algorithm can learn without being explicitly programmed with a conditional logic. It has more options to choose from than the ad hoc algorithm does (functional purposes). This intelligent algorithm may even create new options by self-learning unidentified system states.

5. Define task, formulate procedures, and execute. The right-hand side of the DL describes the execution process, and action automation describes the same. Manual trading and automated trading both require specific technological details of the trading system to

TABLE 1: Function Allocation Mapped on the Four Stages of Automation

Scenario			
Stage 1: Information acquisition	Stage 2: Information analysis	Stage 3: Decision selection	Stage 4: Action
Basket trading (low DOA)			
Partially automated. The trader manually downloads historical market data (**activation**). A MATLAB algorithm organized the data (**observe**).	Automated. The trading software retrieves fundamental information of the short-listed stocks (to buy or sell) from a database (**identify**).	Not automated. The trader decides all trades (**interpret and evaluate**).	Partially automated. The trader determines the parameters of the orders. Once submitted to the market exchange, the orders are proceeded automatically (**define task, formulate procedures, and execute**).
Trend following trading (high DOA)			
Mostly automated. A real-time data feed streamlines data collection (**activation**). The trader typically monitors the market data but is not dependable in the data collection process (**observe**).	Automated. Sophisticated metrics are calculated in real time (**identify**).	Mostly automated. The trading algorithm interprets the situation by examine the metrics with a predetermined criterion. The trader may stop trading (e.g., ”panic button”) but is unable to modify the criterion in real time (**interpret and evaluate**).	Automated. Orders are generated in milliseconds and executed by the market exchange (**define task, formulate procedures, and execute**).

Note. Decision ladder annotations are in bold. DOA = degree of automation.

complete the execution process. The process typically involves multiple steps—for example, to define the direction (generalized functions: “to buy,” “to sell”), to formulate the parameters (physical functions and physical forms), and to decide the destination (physical function: “exchange”).

Mapping the four stages of automation on a base DL provides guidance on representing the function allocation at the task level. “What is more automation” in each stage (Onnasch et al., 2014) can now be represented by annotating the boxes (steps) in the corresponding DL region. Table 1 is a summary of function allocation as seen in the two automated financial trading scenarios that we have been considering. Functions in each stage are annotated with the names of DL steps.

DOA Layering on the Base DL

Similar to our way of representing DOA on the base AH, we used a DOA layering approach for the base DL. Likewise, functions allocated to the automation are shaded, and functions allocated to the user are not shaded. In Figure 5 and 6, we use shaded boxes to represent information processes that are responsibilities of automation (e.g., trading algorithms). Boxes that are not shaded are human information-processing steps, assuming for simplicity that the operator is moving through all the steps of the DL. In this section, we present four cases—two

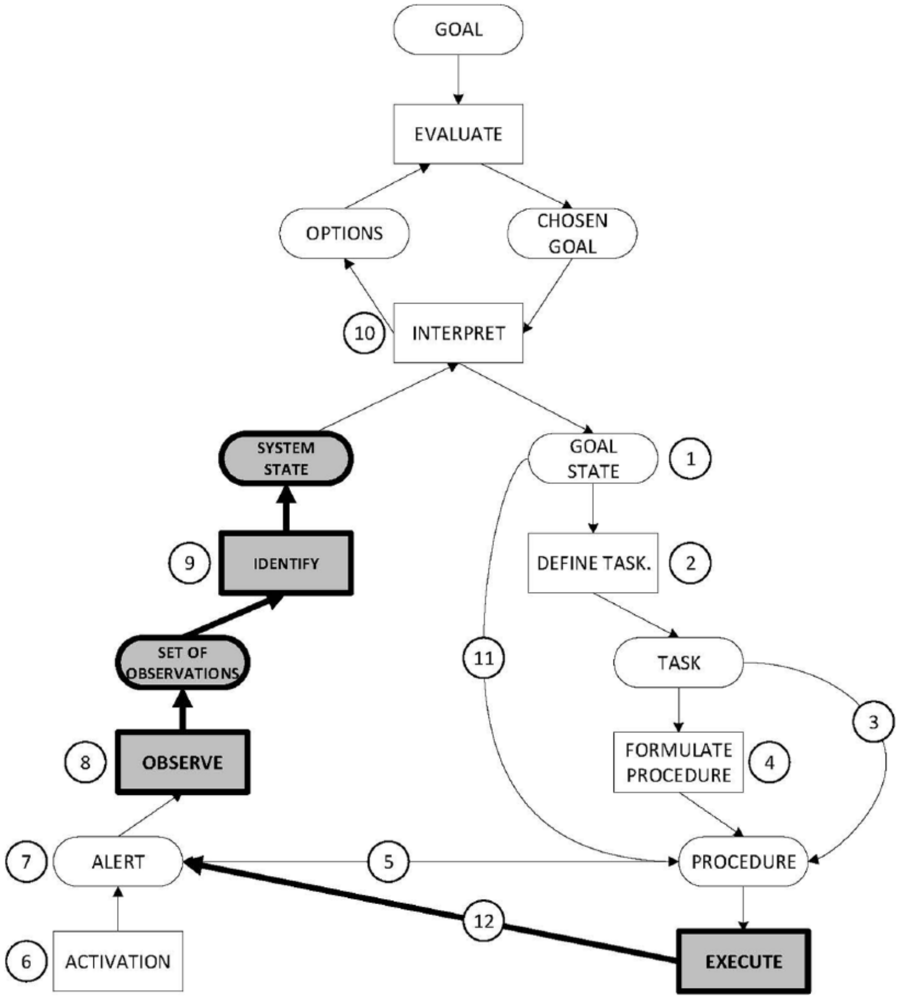

Figure 5. Decision ladder of basket trading (low degree of automation, routine operation).

scenarios (low DOA and high DOA) and two situations (routine operation and unanticipated situation).

Low DOA scenario: The routine operation situation (Case 1). We represent two cases of the low DOA scenario (basket trading): a routine operation DL in Figure 5 and a DL showing unanticipated situations in Figure 6. We have looked at routine and unanticipated situations to show the challenges faced by the trader in intervening in the different automated financial trading scenarios. In each case, a "data analysis and order generation" algorithm is involved in the information acquisition and information analysis stages.

1. *Goal state*. The goal in basket trading is to hold many financial products in certain proportions. The basket of products must be bought or sold simultaneously so that price movements for each product do not alter the portfolio allocation. The basket of products can be stated as follows:

$$\text{basket of products} = \sum_i asset\ allocation_i.$$

Since the proportion of each product is normally preset and required by the trader or the trading institution, the trader begins this routine operation DL with knowledge of a desired goal state.

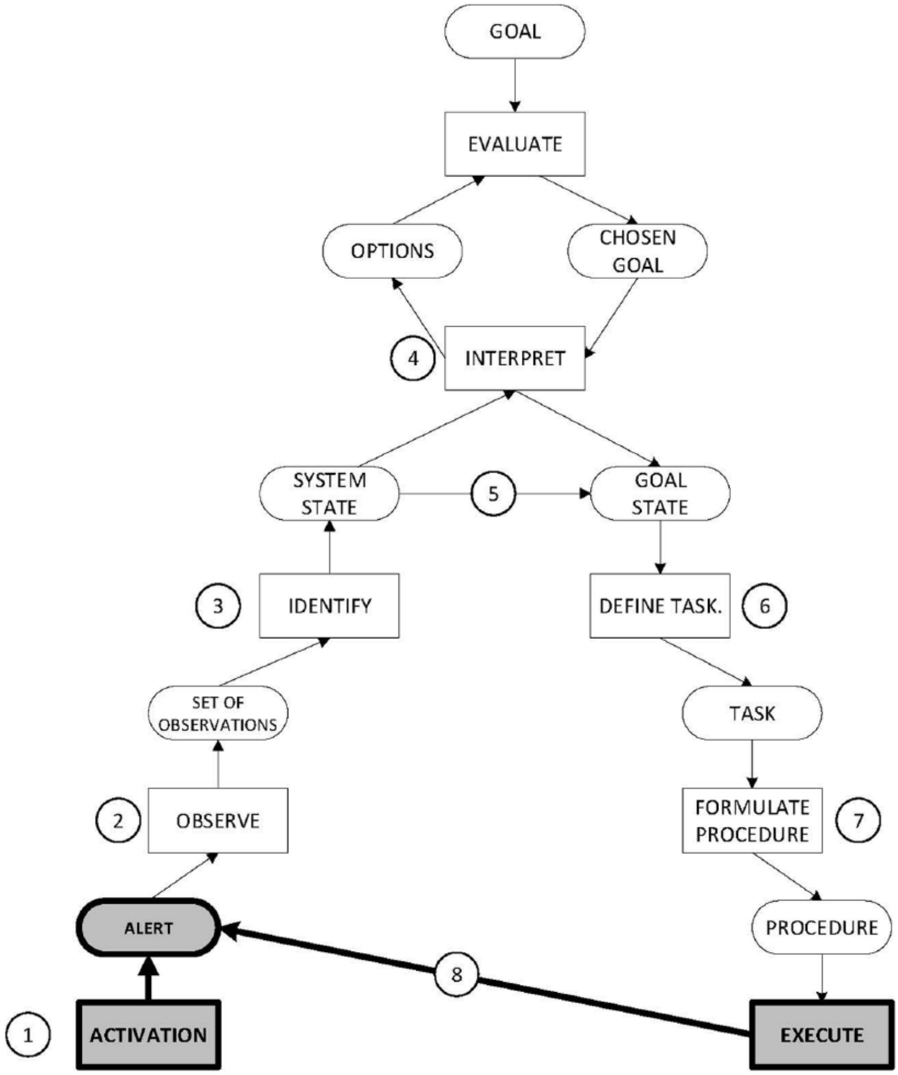

Figure 6. Decision ladder of basket trading (low degree of automation, unanticipated situation).

2. *Define task*. Knowing the goal state, the trader then defines the task that needs to be accomplished in basket trading. For basket trading, this step involves the trader setting up the criteria for shortlisting products that will be traded and deciding what trading action will be conducted.
3. *"Task to procedure" shortcut*. An expert trader may take this shortcut step to transfer knowledge of the task to certain procedure activities, without considering all details of the basket purchase allocation every time.
4. *Formulate procedure*. To generate a complete basket, the trader must consider position limit and other administrative or trading restrictions on each product of the basket. In the futures market, for example, position limit is the highest number of futures contracts that a trader may hold on the premise of deposit. In this case, the trader may fine-tune the basket purchase allocation that does not violate the regulations.
5. *"Procedure to alert" shortcut*. Instead of manually carrying out the task, the trader can take this shortcut step to transfer knowledge of the task to a certain data form that will be later used by the algorithm.
6. *Activation*. The trader must download and submit historical data from the market into the automation at the beginning of each trading day.

7. *Alert*. The resulting alert will indicate to the automation that data are ready for analysis. The data are combined with the desired proportion of each financial product and will be provided to the analyzing program.
8. *Observe*. The algorithm contains a MATLAB script (Chan, 2009) to organize (e.g., sort, rank, index, and select) data according to predefined criteria into suitable formats. The resulting observations contain a shortlist of financial products. The resulting observations also involve orders that will indicate to the automation that data are ready for analysis.
9. *Identify*. The algorithm generates a basket of orders that should lead toward a desired product allocation portfolio.
10. *Interpret*. The basket of orders must be interpreted by the trader before being submitted to the market. In financial trading, many system states are unique and could change in a short period. At this step, the trader must undergo an interpretation of the consequences (Rasmussen, 1974).
11. *"Goal state to procedure" shortcut*. The trader decides to submit all orders. The task and all the procedures have been determined by the trader and the algorithm.
12. *Execute*. The brokerage's trading application (action implementation) submits the basket of orders to the market. The DL ends at this step.

Low DOA scenario: The unanticipated situation (Case 2). The basket trading system can also be operated in an unanticipated mode if it is not possible to execute the basket trade on all products. As a consequence, it will be difficult to hold products in their correct proportions. Alternatively, the trader could make a wrong choice on the financial products or their proportions in the basket. A violent price fluctuation of a single product can nullify all the gains or expose the trader to losses. In this case, the basket trade cannot provide the trader protection against volatility. The unanticipated mode is represented in Figure 6. At a low DOA, the algorithm does not contain a diagnosis feature; therefore, most of the decision making is completed by the trader.

1. *Activation*. The DL starts in automated information acquisition—that is, the brokerage trading software receives quotes from the market exchange via an electronic communication protocol (e.g., Financial Information eXchange). In an unanticipated situation, the resulting alerts contain quantitative data (e.g., unfilled order quantities) and the reason of order rejection (e.g., no financial product definition has been found for the purchase request).
2. *Observe*. The trader must observe the variables and compare them with their respective desired values to assess the fault. For example, the trader must try to probe the reason of order rejection.
3. *Identify*. The trader may identify the cause of the fault and whether the fault is fixable. For example, if illiquidity is the principle problem and cannot be mitigated in a short term, the trader may stop the attempt to rebalance the proportions of the products.
4. *Interpret, evaluate, and reevaluate*. The trader decides what action to take to manage the fault.
5. *"System state to goal state" shortcut*. A reoccurred situation provides knowledge that can accelerate the decision-making process.
6. *Define task*. The trader now defines the necessary task, depending on the type of the unanticipated situation, that can move the system to the correct state.
7. *Formulate procedure*. The trading platform takes over control from the trader. Procedures are formulated according to the decision made by the trader. The trading platform may either continue to purchase financial products in the basket or hold on to the current portfolio.
8. *Execute*. The trading platform submits orders (e.g., buying orders or stop orders). The trader receives a confirmation message from the market exchange.

Essentially, in the unanticipated situation, with the low DOA, the trader must take over the observations and determination of system state, continuing to execute or not as required.

High DOA scenario: The routine operation situation (Case 3). The high DOA scenario (trend following trading) is represented in Figure 7 (routine operation) and Figure 8 (unanticipated situation). In a routine operation mode, a scalping algorithm is first developed by an algorithm developer. At the beginning of each trading day, the trader first downloads data, strategizes with

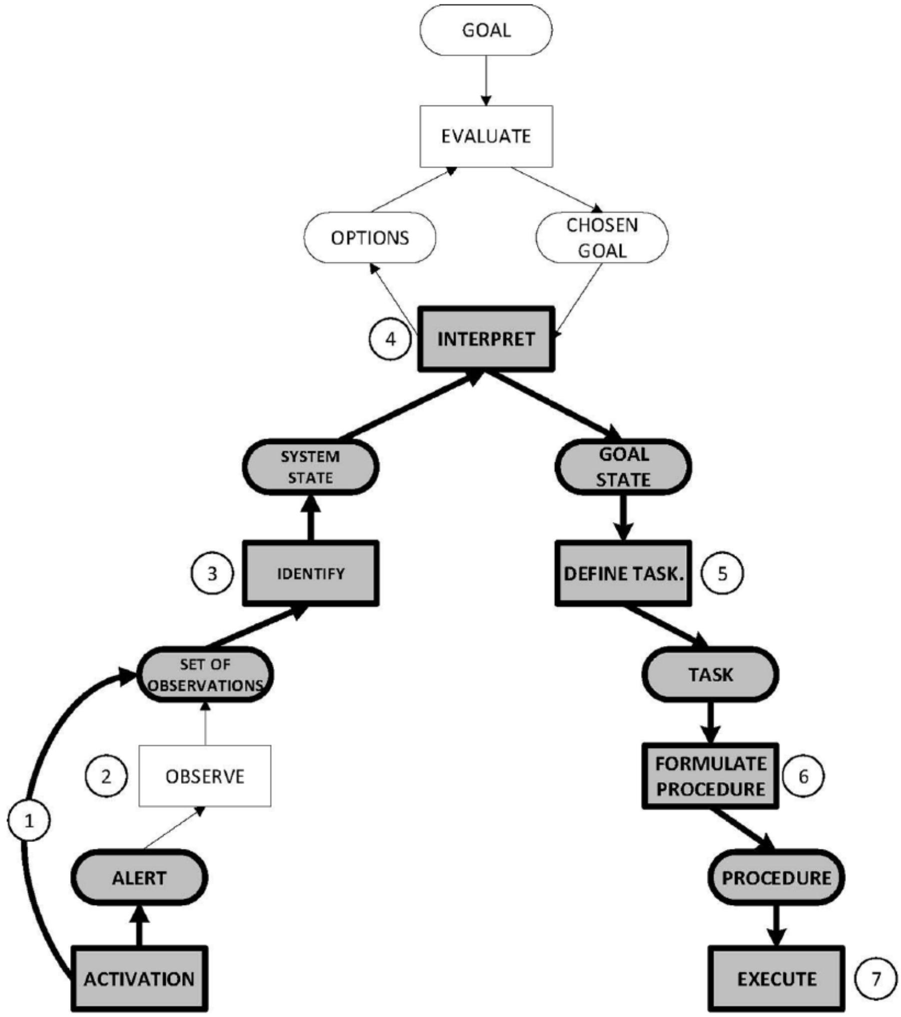

Figure 7. Decision ladder of trend following trading (high degree of automation, routine operation).

other traders and clients, and then starts up various applications, including the scalping algorithm and the trading platform.

1. *"Activation to set of observations" shortcut.* A real-time data feed (information acquisition) receives quotes from the market.
2. *Observe*. Automation at this degree uses a predetermined trading shortlist; therefore, it does not filter the data. The trader may observe the real-time data, but this step is optional, as it does not provide inputs for observations.
3. *Identify*. The quantitative trading algorithm (information analysis) calculates 50- and 25-day SMA and the relative strength index for preselected stocks. Note that this is an automated process, as the variables being calculated, the calculating methods, and the stocks were determined before this trading task started.
4. *Interpret*. For each listed stock, when the 50-day SMA crosses above the 200-day SMA and when the relative strength index in an oversold territory is <30, the quantitative trading algorithm (decision selection) interprets the situation as a buying signal. Automation at this DOA does not provide alternatives. It will not trade if there is no designated trading signal.
5. *Define task*. The scalping algorithm (action automation) determines a buying task when the buying signal occurs.

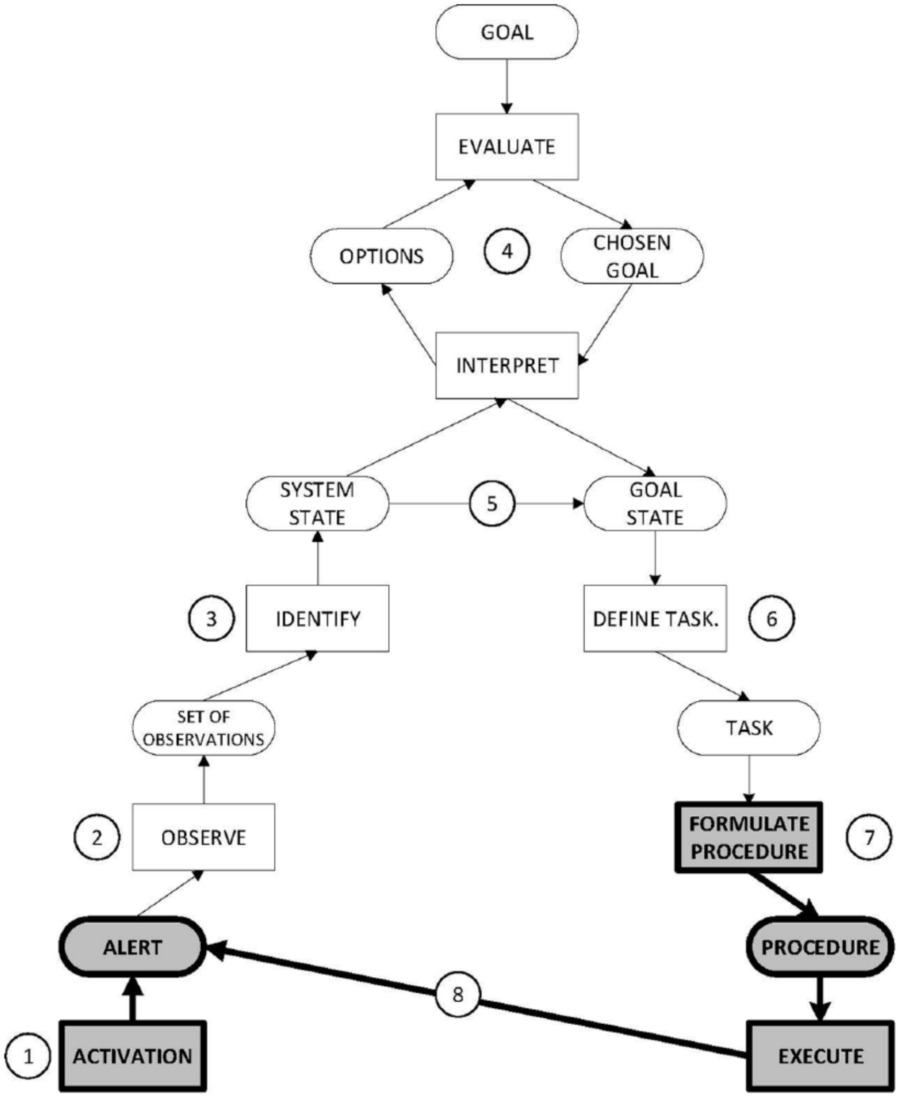

Figure 8. Decision ladder of trend following trading (high degree of automation, unanticipated situation).

6. *Formulate procedure*. The scalping algorithm (action automation) randomizes the size of each order (400 to 800 shares) and determines other order parameters.
7. *Execute*. The scalping algorithm (action automation) places 10 iterations of orders to market, then waits for a confirmation message from the market exchange.

High DOA scenario: The unanticipated situation (Case 4). The "trend following trading" system may face a disturbance and be faced with an unanticipated situation. Possible disturbances are algorithm bugs (e.g., incorrect order quantity), event risk (e.g., political event), and illiquidity. Illiquidity happens during times of low volatility when market price swings in a small range. The lack of liquidity causes a slippage, a difference between "the intended price of a trade and the price at which the trade is really executed" (Investopedia, n.d.). A tremendous loss of liquidity of many financial products, or systemic illiquidity, disturbs the entire market and fails most trading systems in it (e.g., the May 6, 2010, "Flash Crash," Minotra & Burns, 2016; U.S. Commodity Futures Trading Commission & U.S. Securities & Exchange Commission, 2010). The scalping algorithm used in the "trend following trading" system being discussed requires a highly volatile market to enter and exit a trade at will to get a good price for the order fill. During an unanticipated situation, the trader must intervene to take a diagnosis task to understand the situation and try to save the system from the disturbance. The diagnosis task is presented as follows.

1. *Activation*. Just as we discussed before for the case of basket trading in an unanticipated situation, the DL begins with receiving quotes from the market exchange. The brokerage trading software summarizes market information that can contribute to the set of observations.
2. *Observe*. In an unanticipated situation, the trader must observe the collected data. The resulting observations indicate all evidence that a disturbance has happened. For example, to estimate the likelihood of a systemic liquidity risk, observations must be made on illiquidity in multiple stocks and market indexes.
3. *Identify*. The trader now identifies the current state of the trading system and confirms the type and magnitude of the disturbance. The resulting system state also involves knowledge of the control law of the trading strategies.
4. *Interpret, evaluate, and reevaluate*. To generate the knowledge of goal state, it is especially important to evaluate the current system state and justify the efficacy of goal state. This stage is extremely time-consuming and may include additional data-processing activities on top of the DL. For example, when a market crash is observed, traders must be very careful in setting up new trading strategies. The trader must decide whether an algorithm-placed order has been processed by the market exchange.
5. *"System state to goal state" shortcut*. Another constraint to generate a goal state is timing. With the same example of a market crash, sophisticated trading strategies may provide robust market disturbance tolerance (e.g., a certain market condition and trading status can trigger a precaution trading execution), and some of the extreme market conditions can be considered in algorithm design. In this case, there is a state knowledge transfer from the current system state to the goal state. The transfer is illustrated as a leap from system state to goal state.
6. *Define task*. Because the trading system has a high DOA, there are limited options to recover the system from hazardous conditions to compensate for the disturbance. The trader may reconfigure the current trading system to compensate for the disturbance (e.g., modifying an offset setting of the SMA crossover rule). In the case of an irretrievable disturbance, a stop-loss task is decided by the trader.
7. *Formulate procedure*. The scalping algorithm (action implementation automation) determines order parameters.
8. *Execute*. The brokerage trading software (action implementation automation) places new orders to the market and waits for a confirmation message from the market exchange. For example, the scalping algorithm submits a stop order to the market exchange. A confirmation message is then received from the market exchange.

It becomes apparent that in the two high DOA cases—routine operation and unanticipated situation—the trader must interrupt the automation and assume a larger scope of control. Furthermore, because the automation is likely handling small fluctuations well, the problem at hand is likely more complex than usual—for example, a market liquidity change, as discussed. Compared with the low DOA cases, the opportunities to recover are more limited as more information-processing steps are allocated to the automation. The automation can result in rapid executions that can be challenging to interrupt.

DISCUSSION

In this modeling exercise, we proposed a DOA layering approach for conducting two analyses—first the WDA and then the ConTA. For each analysis, we first built a base model (i.e., AH or DL), then mapped function allocation in an additional layer. The base model is similar in many respects to that of an original AH or DL consistent at any DOAs. Functions in the base model can be allocated to any actor (human or automation) and represented in the

DOA layer. Shared allocations could also be included in the DOA layer, though the analyst may want to differentiate between shared allocation approaches. The DOA layer adds a new dimension to the base model, showing how human and automation work collaboratively at a certain DOA. The DOA layering approach can be used for representing function allocations at the domain level and the task level. First, the presented base AH example shows that the physical and functional structures are consistent in the basket trading system and the "trend following trading" system. The DOA layer suggests that function allocations are different in the two trading systems, depending on the system's DOA. In the DL examples, we analyzed two tasks—routine operation and unanticipated situation—of the basket trading system and the "trend following trading" system. Second, the base DL as a template (Vicente, 1999) was augmented with four regions to show the four stages of automation. The DOA layer enriches the base DL with more features, such as automated information-processing steps, stages of knowledge, and shortcuts.

The following discussions focus on a comparison of the DOA layering approach and how automation was previously modeled with CWA, including what more design implications the DOA layering approach might have for designing support for automation than existing approaches. Last, we discuss implications of applying the DOA layering approach to adaptive automation.

Comparing the DOA Layering Approach With the Dual-Model Approach

In this section, we compare the DOA layering approach to how automation was represented in the CWA literature. In most cases, the CWA literature explicitly focusing on modeling automation has taken a *dual-model* approach, with one model showing nonautomated systems and a separate model showing automated components. We first introduce the dual-model approach and its origin, then conduct a comparative analysis of the DOA layering approach and the dual-model approach. The objective of this analysis is not to draw a conclusion on which approach is superior to the other. Instead, we suggest that either approach has its own application depending on the type of system and problem being modeled. The applicable occasions of the two approaches are discussed, suggesting when to develop the DOA layered model and when to develop the dual model.

Dual-model approach. Typically, automation and function allocation requirements are explained in the social and organizational analysis of CWA, after the WDA and the ConTA are completed (Vicente, 1999). The dual model is a relatively new approach formally introduced by Mazaeva and Bisantz (2007), in a digital single-lens reflex camera analysis study. Mazaeva and Bisantz suggested that automation should be explicitly modeled at the WDA and ConTA phases, through AH and DL tools. We found that the two aspects of the dual-model approach—the dual-model AH and the dual-model DL—have somewhat different origins.

1. *Dual-model AH.* The original AH proposed by Rasmussen (1986) modeled automation as work domain components at the lower AH levels, which are physical function and physical form. It seems to be an appropriate modeling decision in the various examples of automated systems given by Rasmussen. These examples include a washing machine (pump and valve function, configuration and weight, and size), manufacturing plant (physical functioning of equipment and machinery), and computer system (electrical function of circuitry). Burns, Bisantz, and Roth (2004) suggested an extension to the original AH to represent automation with WDA. Burns et al. comparied decisions made on modeling naval sensor systems. They proposed that developing dual AH models, in which one AH represents the system being controlled and another AH represents the automation, may "help specify information needs for those responsible for automation monitoring and control." Inspired by Burns and colleagues' work, Mazaeva and Bisantz (2007) provided a detailed example of the dual-model AH, introducing a "camera AH" to show nonautomated camera components that are being controlled and a separate "automation AH" to show automated systems that allow for automated movement and exposure control of the camera's components. Their dual models also represent interconnections between levels of the two AHs.

2. *Dual-model DL.* To our knowledge, the first introduction of the dual- or multiple-model approach is Rasmussen and Goodstein's work (1987). It was pioneering work, using three DLs to represent cooperative decision making in a nuclear reactor control domain among different actors, including a human designer, an operator, and a computer. Each DL reveals a different subtask, and together they complete a control task (e.g., operator intervention during an accident). Similarly, Mazaeva and Bisantz (2007) developed three DLs, representing interrelated control tasks distributed across the automation, the photographer, and the designer represented.

Occasions where the dual-model approach fits. Table 2 briefly summarizes the differences between the dual-model approach and the DOA layering approach.

The dual-model approach is a successful first attempt to explicitly represent automation within a CWA model. To understand the applicability of the dual-model approach, note that Rasmussen and Goodstein's approach (1987), the origin of the dual-model approach, was initially proposed for supporting supervisory control system design. Supervisory control is associated with an intermediate DOA (Sheridan & Verplank, 1978). The system boundary is clear for analyzing a supervisory control system, whereby the automation takes a task performer's role in the closed inner loop, while the operator manipulates control parameters in an outer loop (Sheridan, 2011). Similarly, Mazaeva and Bisantz's (2007) AHs and DLs are exclusive representations of decision process allocation within the ongoing supervisory control. They looked at a digital single-lens reflex camera, a commercial product whose DOA has already been decided. In other words, the dual-model AHs and DLs are constrained by a certain DOA.

From Rasmussen's (1986) example and Mazaeva and Bisantz's (2007) example, we can see that the dual-model approach is an appropriate approach to model with a fixed DOA and to analyze and understand existing automated systems. This finding echoes the work of Burns and colleagues (2004), who pointed out that the automation AH is "perhaps created later in the analysis, once the levels of automation have been specified." In this case, guiding automation design is not the primary objectives of the analysis. Instead, analysts may focus on addressing multiple control tasks and strategies to represent sophisticated interactions between human and automated behaviors, at a predetermined DOA. Since the dual-model approach develops a more explicit model of the workings of that automation, the dual-model approach is a good choice where operators must diagnose or fix the automation itself.

Occasions where the DOA layering approach fits. There are certain occasions where the DOA layering approach fit better than the dual-model approach.

1. *Modeling systems with a variable DOA.* Mazaeva and Bisantz (2007) modeled a fixed DOA system; the dual models lack flexibility of analyzing domains of a variable DOA. Analysts using the dual-model approach may encounter a scalability issue while applying the dual-model approach to guide automation design in personal and property safety systems (e.g., aviation, process control, finance) significantly more complex than a camera system. However, the automated financial trading system that we analyzed is an example of intelligent automated systems with a consistent physical structure and a variable DOA. The proposed DOA layering approach is a single-model approach. In the presented AH examples, a consistent physical structure is shared between the human and the automation. Likewise, in the presented DL examples, the same control task is shared between the human and the automation. The DOA layered models have simplicity in how automation is modeled. Our approach may prove useful in representing more coordinated human-automation interaction, by leaving more flexibility in modeling multiple system modes (e.g., routine operation and unanticipated situations).
2. *Incorporating "stages and levels of automation" model.* CWA was built for understanding complex automated systems. *Automated* is an important aspect of sociotechnical systems (Vicente, 1999). A considerable interest has grown in the human factors community around the theme of how to model the DOA (e.g., Sheridan & Verplank, 1978;

TABLE 2: A Comparison of the Dual-Model Approach and the DOA Layering Approach

	Dual-model approach	DOA layering approach
Basic concepts	Allocate user and automated system functions to separate AHs. Allocate user and automated procedures to separate DLs.	Allocate user and automated system functions to separate layers in the same AH. Allocate user and automated procedures to separate layers in the same DL.
Deliverables	User model (AH and DL). Automation model (AH and DL).	Base model (AH and DL). User layer (AH and DL). Automation layer (AH and DL).

Note. DOA = degree of automation; AH = abstraction hierarchy; DL = decision ladder.

Parasuraman et al., 2000). Among the previous examples discussing how to explicitly model automation in CWA, the "stages and levels of automation" model has not been well utilized in the CWA literature. The DOA layering approach takes the first step to fill this gap, by transforming DOA knowledge to CWA models. We found that ConTA could play an important role in describing how information-processing activities are allocated to the user and the automation.

3. *Supporting automation design*. The DOA layer, layering on the base DL, is a supplement to the "stages and levels of automation" model. The DOA layering approach supplements the "broad-brush" description of levels of automation (Pritchett, Kim, & Feigh, 2014) by enabling two important features within DL, each demonstrating a shortcut between information-processing steps: (1) the leaps and shunts that an expert takes, in the case of human information processing; (2) the bypassing of a nonautomated step, in the case of automated information processing. The former feature is inherent from the original DL and is still available to analysts using the DOA-layered approach. The latter feature describes alternative routes of information processing, implying opportunities of human operators.

Implications for Design

The DOA layering approach makes unique contributions to automation design, not only designing automation displays but also deciding stages and levels of automation. We discussed two possible design implications: designing ecological automation displays with DOA and constraint-based procedure supports and deciding stages and levels of automation. We used the presented AHs and DLs as examples.

Implication for display design: Designing ecological automation displays. The ecological interface design approach requires user interface designers to first conduct an information analysis to extract information from a completed AH (Burns & Hajdukiewicz, 2004). Next, this information should be organized as a list of variables for representational design, with constraints from the work domain.

The DOA layering approach allows user interface designers to capture variables from the base AH and constraints from the base AH and the DOA layer. For example, in the "trend following trading" AH, a functional purpose of the system is to achieve a maximum rate of revenue in trading. This function could be described by revenue run rate, a metric for predicting future financial performance based on the current financial information. The constraint of this metric is decided by the technical limitation of the trading system and can be found in the DOA layer. Allocating this functional purpose to automation means that the trading system is running in real-time in a day trading setting. Therefore, a short duration (milliseconds or seconds) of this metric must be calculated and monitored by the automation, as the trader is incapable of monitoring the rate of revenue in an extremely short duration. However, at the physical form level, variable and fixed cost functions are allocated to the trader. According to the base AH, the two types of costs are constrained by a certain currency type of the trading market. Other constraints are related to the trader only, not the automation. They are trader-specific information, such as the trader's personal financial status, indicating that the trader is ultimately responsible for cost control in a "trend following trading" system.

More variables and constraints can be seen from the means-ends relationships on the base model, as well as the DOA layer overlaid. For example, "position" at the physical function level of the base AH connects to "market price," "order price," and "position price" at the physical form level, suggesting that market and portfolio are two interrelated sides and an integrated market-portfolio display may support direct perception of information from both sides. With a high DOA layer, such a relationship remains consistent, but "position" is allocated to the trader, and the price-related functions are allocated to the automation. This allocation suggests that, in a trend-following system, although these price-related functions are represented on the display with appropriated constraints, the trader does not take control of these functions. Therefore, additional visualizations may be provided to the trader to understand how automation processes these functions.

Implication for automation design: Determining automation stages and levels. Another implication of the DOA layering approach is that this approach could fit into the framework for automation design proposed by Parasuraman et al. (2000) to help determine automation stages and levels. The "stages of automation" model, an important "starting point for considering what types and levels of automation should be implemented in a particular system," provides "a simple guide for automation design." The framework suggested that automation design should begin with identifying what class of functions should be automated. The automation designers then apply evaluative criteria (e.g., automation reliability and situation awareness) and recommend "particular levels of automation for each of the four types of automation."

We believe that fitting the DOA layering approach to an existing automation design framework could supplement the "stages and levels of automation" model rather than replace it. We suggest that automation designers may use the DOA layering approach at an early phase of automation design, before applying evaluative criteria, to help them determine what stages and levels of automation are appropriate for the system. The base DL represents the four functional domains on a DL, providing an easy start point for automation designers to develop a conceptual design estimation. We hope trading algorithm developers may consult with the base DL in the future to decide which algorithm to use: an intelligent algorithm (i.e., decision automation) or an order-placing script (i.e., action automation). Automation designers must also justify the use of a certain stage of automation. At a subsequent stage of automation design, automation designers must decide what level of automation should be developed within each functional domain (Parasuraman et al., 2000). The DOA-layered DL provides richer information than the "stages and levels of automation" taxonomy. The DL shows not only what human or automation functions should be applied within each stage (shades) but also what aspects of human interactions with automated systems should be considered (shortcuts).

The DOA layering approach could help with understanding and design for modern, intelligent automation. The DOA layering approach echoes a recent suggestion by Sheridan (2017), suggesting that modern automation is hierarchical in the same way as the human work competencies. If modern automation is hierarchical, then automation competencies can be modeled by the skill-rule-knowledge (SRK) taxonomy, the last phase of CWA, which has been used in the literature only to model human work competencies. Sheridan gave hints for identifying an SRK for automation: (1) skills of the automation are continuous actions triggered by the laws of physics (e.g., the spinning of steam turbine) but are conditioned through commands of an automated agent (e.g., a programmable logic controller) or human agent (e.g., an operator); (2) supervisory control and artificial intelligence go beyond the traditional realm of classic feedback control and invoke the "rule" or "knowledge" level on the hierarchy of SRK. An "If *x*, then *y*, else *z*" logic forms a stored rule to invoke designated human activities (e.g., an action recommendation system) or automation activities (e.g., action automation); (3) automation based on the knowledge level is rare but becomes possible in machine learning–based systems, such as the IBM Watson computer.

The implication of the SRK for automation is that automation may use all stages and levels of information processing. By modeling the DOAs on the AH and DL, it can clearly be seen that in the higher DOA situation, functions at the higher AH levels (e.g., functional purpose and abstract function, in the high DOA AH) and information-processing steps on top of the DL (e.g., interpret, in the high DOA, routine operation DL) are allocated to the automation, which may present knowledge-based automation (cf., Rasmussen, 1986). The interconnections of the presented AH and DL examples and SRK for automation suggest a future extension of DOA layering, which layers function allocations in other phases of CWA to support automation design.

Implications for Modeling Adaptive Automation

The DOA layer on the DL may help the analyst model DOA shifts, shedding some light on how to model adaptive automation in the future. For example, Table 3 presents two cases of DOA shifts—a DOA increase case and a DOA decrease case—based on the high DOA scenario. The table shows that a DOA shift can occur at any DL step, as any DL step (box) can be shaded (i.e., functions reallocated to automation) or not shaded (i.e., functions reallocated to the human). However, DOA shifts can be frequent, as algorithm development is an extremely flexible process depending on traders' expertise and preference. It is also an iterative process, with each iteration starts from developing, back testing, to live trading. At this stage, the DOA layering approach portrays the relationship between human and automation functions at a task level, and we hope that it grows into a potentially useful approach for modeling adaptive automation.

CONCLUSION

Information systems should support human-automation coordination (e.g., either human or automation must seamlessly switch between responsibilities). CWA helps the development of "simple qualitative models" (Sheridan, 2017) that can be represented by graphical interfaces. An adoption of function allocation models, such as the "stages and levels of automation" model to CWA, could provide a new design opportunity. Yet, this approach has been not well developed. We attempted to fill this gap by proposing a DOA layering approach, layering DOA on AH and DL to express domain- and task-level function allocations, respectively. This paper is an extension to two earlier versions in the *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* (Li et al., 2015, 2016).

Automated financial trading, a domain rarely explored by the human factors community, served as an appropriate example in this modeling exercise. Automation in financial trading is versatile in terms of the various stages and levels of automation involved; the highly coupled relations among traders, infrastructure, and trading algorithms; and the unpredictable dynamics in the environment. Two scenarios of financial trading are provided in this paper, and each has a unique DOA. New models in the context of automated financial trading were developed with extended AHs and DLs with DOA layers. In each case, a base model was first created, followed by mapping two scenarios (low DOA and high DOA) onto the base model. In particular, we correlated the "stages and levels of automation" model (Parasuraman et al., 2000) to the DL, whereby DL steps were organized into four stages. This paper is the first to propose a DOA layering approach and the first to comprehensively use CWA and the "stages and levels of automation" model to model automated trading.

This paper provides useful insights to the debate of using a single- or dual-model approach to model automated systems. The DOA layering approach extended the flexibility of the single-model approach by representing the DOA, and it echoes Sheridan's (2017) recent homage to Rasmussen's frameworks (e.g., AH) for their robustness and applicability to behaviors of humans or highly intelligent automation. Future works include how to model adaptive automation with the template-layering approach. This paper also corroborates recent suggestions of Borst et al. (2015) on providing more automation status on ecological displays to support human-automation

TABLE 3: Example Reasons for DOA Shift per DL Step in Trend Following Trading

	Reasons for DOA shift	
DL processing steps (in financial trading terms)	DOA increases	DOA decreases
Activation by normality and abnormality in market or portfolio.	Automated signal detection is capable (e.g. timely tick data in shorter duration; Level II data); impulse control	Technology is unavailable due to high costs or lack of work competence, distrust in technology (e.g., concerns with latency of the data), and obsessive financial market monitoring.
Observe the dimensions of the issue.	High computing power is available for real-time pattern generation	Countervailing trading philosophy (e.g., fundamental analysis is favored over real-time technical analysis).
Identify the current state.	High computing power is available for real-time pattern recognition; system state can be quantitatively modeled	Concerns with latency in pattern recognition (e.g., unavoidable delay in automated executing).
Interpret the ambiguity of historic and current states, as well as the consequences of future states; **evaluate** the current state with a goal from a higher level of abstraction.	Artificial intelligence advances; no or little ambiguity in the current status the current market condition is predicted; historic market data is accessible and understandable by the trading algorithm; prediction model is reliable	Automation is not capable to interpret or is believed misinterpreted the current status; market condition is abnormal; the current status is interpretable, but the consequences of future states are not acceptable (e.g., risk of spoofing, see *N. D. Ill. v. Sarao*, 2015).
Define task in financial trading (e.g., buy or sell).	Indispensable in some high-frequency trading systems (Chan, 2009); adequate knowledge of automated trading and high-performance programming.	Complexity and cost are not acceptable.
Formulate procedures; in another word, generate orders with appropriate arguments (e.g., order quantity, order price, and target financial product).	Indispensable in some high-frequency trading systems.	
Execute an order in the market.	Indispensable in some high-frequency trading systems.	Lack of knowledge in high-performance programming but semiautomated alternative.

Note. DL annotations are in bold. DOA = degree of automation; DL = decision ladder.

coordination. We will further examine the design implications in an experimental study.

ACKNOWLEDGMENTS

This project is financially supported by a Natural Sciences and Engineering Research Council of Canada grant (CRDPJ 445965-12). Special thanks to Quantica Trading Inc. for providing the venue for the participant observation. We thank Drew Coles, Roni Hoffman, Travis Felker (Quantica Trading Inc.), Dr. Rui Hu (Microsoft Corporation), and Xian Wang (Shenzhen Platinum Venture Capital Ltd.) for serving as the subject matter experts. We also acknowledge the comments of our associate editor Dr. Emilie M. Roth and three anonymous reviewers.

REFERENCES

Achonu, J., & Jamieson, G. A. (2003). Work domain analysis of a financial system: An abstraction hierarchy for portfolio management. In *Proceedings of the 22nd European Annual Conference on Human Decision Making and Control* (pp. 102–109). Linköping, Sweden: Cognitive Systems Engineering Lab.

Bainbridge, L. (1983). Ironies of automation. *Automatica*, *19*(6), 775–779.

Bollinger, J. (2001). *Bollinger on Bollinger Bands*. New York, NY: McGraw-Hill.

Borst, C., Flach, J. M., & Ellerbroek, J. (2015). Beyond ecological interface design: Lessons from concerns and misconceptions. *IEEE Transactions on Human-Machine Systems*, *45*(2), 164–175.

Burns, C. M., Bisantz, A. M., & Roth, E. M. (2004). Lessons from a comparison of work domain models: Representational choices and their implications. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *46*(4), 711–727.

Burns, C. M., & Hajdukiewicz, J. R. (2004). *Ecological interface design*. Boca Raton, FL: CRC Press.

Chan, E. (2009). *Quantitative trading: How to build your own algorithmic trading business*. Hoboken, NJ: Wiley.

Davis, M., Kumiega, A., & Van Vliet, B. E. (2013). Ethics, finance, and automation: A preliminary survey of problems in high frequency trading. *Science and Engineering Ethics*, *19*(3), 851–874.

Ellis, C. A., & Parbery, S. A. (2005). Is smarter better? A comparison of adaptive, and simple moving average trading strategies. *Research in International Business and Finance*, *19*(3), 399–411.

Fabozzi, F., Focardi, S. M., & Jonas, C. (2011). High-frequency trading: Methodologies and market impact. *Review of Futures Markets*, *9*, 7–38.

Furukawa, H., & Parasuraman, R. (2003). Supporting system centered view of operators through ecological interface design: Two experiments on human-centered automation. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *47*(3), 567–571.

Hu, R., & Watt, S. M. (2014). An agent-based financial market simulator for evaluation of algorithmic trading strategies. In *Proceedings of the Sixth International Conference on Advances in System Simulation* (pp. 221–227). Nice, France: International Academy, Research, and Industry Association.

Iati, R. (2009). *The real story of trading software espionage*. Retrieved from http://www.advancedtrading.com

Investopedia. (n.d.). *Slippage definition*. Retrieved from http://www.investopedia.com/terms/s/slippage.asp

Kaber, D. B., Riley, J. M., Tan, K. W., & Endsley, M. R. (2001). On the design of adaptive automation for complex systems. *International Journal of Cognitive Ergonomics*, *5*(1), 37–57.

Leaver, M., & Reader, T. W. (2016). Human factors in financial trading: An analysis of trading incidents. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *58*(6), 814–832.

Li, Y., Burns, C., & Hu, R. (2015). Understanding automated financial trading using work domain analysis. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *59*(1), 165–169.

Li, Y., Burns, C., & Hu, R. (2016). Representing stages and levels of automation on decision ladder: The case of automated financial trading. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *60*(1), 328–332.

Mazaeva, N., & Bisantz, A. M. (2007). On the representation of automation using a work domain analysis. *Theoretical Issues in Ergonomics Science*, *8*(6), 509–530.

McAndrew, C., & Gore, J. (2013). Understanding preferences in experience-based choice: A study of cognition in the "wild." *Journal of Cognitive Engineering and Decision Making*, *7*(2), 179–197.

McIlroy, R. C., & Stanton, N. (2015). Ecological interface design two decades on: Whatever happened to the SRK taxonomy. *IEEE Transactions on Human-Machine Systems*, *45*(2), 145–163.

Minotra, D., & Burns, C. M. (2016). Understanding safe performance in rapidly evolving systems: A risk management analysis of the 2010 US financial market Flash Crash with Rasmussen's risk management framework. *Theoretical Issues in Ergonomics Science*. Advance online publication. doi:10.1080/1463922X.2016.1254837

N. D. Ill. v. Sarao, United States District Court Northern District of Illinois Eastern Division (2015). Retrieved from http://www.justice.gov/sites/default/files/opa/press-releases/attachments/2015/04/21/sarao_criminal_complaint.pdf

Onnasch, L., Wickens, C. D., Li, H., & Manzey, D. (2014). Human performance consequences of stages and levels of automation: An integrated meta-analysis. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *58*(3), 476–488.

Parasuraman, R., Sheridan, T. B., & Wickens, C. D. (2000). A model for types and levels of human interaction with automation. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, *30*(3), 286–297.

Pritchett, A. R., Kim, S. Y., & Feigh, K. M. (2014). Modeling human-automation function allocation. *Journal of Cognitive Engineering and Decision Making*, *8*(1), 33–51.

Rasmussen, J. (1986). *Information processing and human-machine interaction: An approach to cognitive engineering*. New York, NY: North-Holland.

Rasmussen, J. (1974). *The human data processor as a system component: Bits and pieces of a model* (Report No. Riso-M1722). Retrieved from Technical Univeristy of Denmark website: http://orbit.dtu.dk/files/56551783/ris_m_1722.pdf

Rasmussen, J., & Goodstein, L. P. (1987). Decision support in supervisory control of high-risk industrial systems. *Automatica*, *23*(5), 663–671.

Sarter, N. B., & Woods, D. D. (1995). How in the world did we ever get into that mode? Mode error and awareness in supervisory control. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *37*(1), 5–19.

Sheridan, T. B. (2011). Adaptive automation, level of automation, allocation authority, supervisory control, and adaptive control: Distinctions and modes of adaptation. *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans*, *41*(4), 662–667.

Sheridan, T. B. (2017). Musings on models and the genius of Jens Rasmussen. *Applied Ergonomics*, *59*, 598–601.

Sheridan, T. B., & Verplank, W. L. (1978). *Human and computer control of undersea teleoperators*. Retrieved from the Defense Technical Information Center website: http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA057655

Sundström, G., & Hollnagel, E. (Eds.). (2011). *Governance and control of financial systems: A resilience engineering perspective*. Farnham, UK: Ashgate.

U.S. Commodity Futures Trading Commission & U.S. Securities & Exchange Commission. (2010). *Findings regarding the market events of May 6, 2010*. Retrieved from https://www.sec.gov/news/studies/2010/marketevents-report.pdf

Vicente, K. J. (1999). *Cognitive work analysis: Toward safe, productive, and healthy computer-based work*. Boca Raton, FL: CRC Press.

Wickens, C. D., Li, H., Santamaria, A., Sebok, A., & Sarter, N. B. (2010). Stages and levels of automation: An integrated meta-analysis. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *54*(4), 389–393.

Yeti Li is a PhD candidate in the Department of Systems Design Engineering at the University of Waterloo, Canada. He completed his master of applied science in systems design engineering at the University of Waterloo in 2013. He is a human factors researcher with an interest in cognitive work analysis, ecological interface design, and human-automation interaction. Yeti served as the president of the Human Factors and Ergonomics Society–University of Waterloo Student Chapter in 2014.

Catherine M. Burns is a professor in the Department of Systems Design Engineering at the University of Waterloo, Canada, where she directs the Advanced Interface Design Lab and the Centre for Bioengineering and Biotechnology. She is a fellow of the Human Factors and Ergonomics Society. Her research is in human factors engineering, where she is well known for her work in cognitive work analysis, ecological interface design, and the development of decision support systems. In this area, she has contributed more than 200 publications and is the coauthor of seven books. Catherine's recent research projects have been exploring multimodal displays for unmanned aerial vehicle ground control stations, decision-making support for automated financial trading, and support for improved medication management.