Expanding Functional Analysis to Develop Requirements for the Design of the Human-Computer Interface

Brian McKenna, James Gualtieri, & William Elm
Resilient Cognitive Solutions, LLC

Abstract

The purpose of this paper is to provide a framework for extending the functional analysis of a system to the human decision-making agents in order to improve overall system effectiveness. The subsystems needed to accomplish the goals of the system can often be found through a functional analysis. However, the different subsystems need to be completely integrated in order for the system to be effective. This integration is often problematic within the Human-Computer Interface (HCI), where the operator is reliant on the information provided in order to have an understanding of what is occurring within the work domain. As a functional analysis helps to define the goals to be accomplished and the subsystems that accomplish them, it follows that expanding the depth of the functional analysis will help integrate the human operator and technology through the HCI to make a more effective Joint Cognitive System (JCS). The functional analysis leads to the creation of more insightful design requirements for the HCI that directly link the work that the user must accomplish and the information needed to complete this work – making it truly user-centered design. An example of how the functional analysis can be applied to a system is provided, comparing an HCI developed with typical design requirements to an HCI developed from an expanded functional analysis.

Introduction

As systems are created to satisfy goals within a work domain, systems need to be designed to best satisfy these goals. Every decision about the shape, structure, layout, material, and even software must be made in the context of how it supports the satisfaction of goals within the domain. The Human-Computer Interface (HCI) must be designed in this context as well. An HCI needs to present a decision-effective visual representation of information so that the user will be able act correctly and decisively to satisfy the goals that have been laid out. Because of the role system goals play in shaping HCI design, it is imperative that the correct system goals are identified during the requirements analysis.

There are a number of ways that system goals may be derived (e.g., through customer requests or through the identification of current system deficiencies); functional analysis of the work domain is one such method that has been effective. The use of a functional analysis impacts the identification of system goals in several ways. First, it allows the system designer to identify *ALL* goals needed for successful operation within the work domain. Although customers and users may offer a detailed set of features the system should have, identifying a complete set of truly decision effective goals for the system a priori has been notoriously vexing to accomplish. Completing an analysis of the domain, however, is a structured process that is specifically tailored to the identification of the necessary goals of the domain. It will also uncover goals for systems to be used in new work domains, a necessity for designing innovative systems. A second implication of functional analyses is that they allow for the identification of relationships between goals. In many cases, the satisfaction of one goal will be dependent on the completion of a number of other goals; understanding these relationships is important for effective system design.

Finally, the functional analysis also helps to define the subsystems that are needed within a domain for the completion of a task. Accomplishing an assortment of goals is often best done through a wide variety of components. The components can be specialized so that each task is completed more effectively and efficiently. As each task is completed more effectively, the system as a whole becomes more productive. The identified functions can also be mapped to physical components to ensure that each function is supported (Bahill & Dean, 2006).

However, distributing the tasks to be accomplished to different physical subsystems comes at a cost. As the satisfaction of a goal is often dependent on the satisfaction of other goals, subsystems will end up being dependent on one another as well. While a subsystem can accomplish a task in isolation, it is still completing the task for a purpose: it must satisfy the demand placed upon it by a supported subsystem. If the subsystem does not provide this support as needed, then the completion of the task will have been for naught. Things get even more complicated as subsystems often have to provide support to more than one other subsystem at the same time. The higher the number of systems a subsystem must support, the more likely that the demands being placed on that subsystem will conflict. Goal confliction - having to satisfy two divergent goals at the same time - is a major source of error within systems (i.e., the confliction is not adequately represented in the display space) which often leads to disastrous results (Woods & Hollnagel, 2006).

Because of the tight relationships between subsystems, system integration becomes a crucial part of the system design process. Work needs to be completed efficiently both within a subsystem as well as across the network of subsystems. **The system will not be as effective as the aggregation of subsystem effectiveness, but, instead, will only be as effective as the integration of the subsystems allows.**

The integration through the HCI creates a human/technology team which must act as co-agents to achieve goals and objectives in a complex work domain. Both the human operators and technology have roles in completing goals within the work domain – together, they are responsible for the entire network of goals. Therefore, the human and technology should be considered a single unit, a Joint Cognitive System (JCS; Hollnagel & Woods, 2005). They must work cohesively in order to successfully satisfy all of the system's goals. The ability to work together is highly dependent on communication between the two. Within a JCS, this communication occurs through the HCI. **However, from the JCS perspective, the HCI is not simply a device to exchange information between the human and technology; the HCI is the way that the work domain is conveyed to the user and the method by which the operator can make actions in the world (McKenna, Gualtieri, & Elm, 2006).**

A key to good integration is that information must be passed between subsystems (Bahill & Dean, 2006). Unlike for most subsystems, however, for the HCI, "information passing" is not completed through a data API, but rather through a representation on the screen to the perceptual and cognitive processing of the operators. In order for operators to successfully work within the JCS, they must have the correct information on the HCI, presented in a way that promotes understanding of the work domain (and the information presentation should take advantage of the known perceptual and cognitive abilities of people). In doing so, the cognitive work the operator needs to perform will be reduced, making it easier for the JCS to complete the identified goals.

The design requirements that are used to guide the HCI design must provide enough detail so that this information can properly be captured on the screen. In doing so, the HCI will allow the operator to work more effectively with the technology within the JCS. A good way to create such requirements for the HCI is through an extension of the functional analysis. **As functional thinking helped to define the goals that are to be accomplished, as well as the subsystems which can be used to satisfy these goals, it follows that expanding the depth of the functional thinking will help define the design requirements that will allow for a better integration of the human operator and technology through the HCI.**

The identification of the system goals must only be the first step in the creation of the HCI. The analysis must continue in order to determine what work needs to be done to satisfy the goals and the information that is necessary to complete the work. Only then can design requirements be developed that guide the HCI design so that it properly supports the human operator within the JCS, ensuring the goals of the domain can be satisfied. Otherwise, the accidents that have plagued many domains [including: nuclear power plants (e.g., Three Mile Island); aviation (Strasbourg Airbus crash; see: Hourizi & Johnson, 2001); and tele-operations (e.g., UAV crashes)] will continue to happen.

Applied Cognitive Work Analysis – a method for developing design requirements for the HCI

The Applied Cognitive Work Analysis (ACWA) is a process through which design requirements (and the resulting HCI representations) are developed, based on a cognitive analysis of the work domain. A key tenet of ACWA is that, in order to develop an HCI that will effectively support the operator, it must be based on the fundamentals of the domain of practice and the demands it imposes on domain practitioners. The resulting HCI will have a high correspondence that provides a setup for cognitive work in context by using the relationships, context, changes, events, and contrasts that give data meaning to organize data (Elm, Gualtieri, Potter, Tittle, & McKenna, in press). Developing requirements for the HCI through the functional analysis produces truly revolutionary display concepts that complement the human user and allow the JCS to successfully complete work in the domain.

The ACWA process begins with a functional analysis of the work domain in order to determine the goals for which the system is responsible. The analysis captures the relationships between goals, identifying interdependencies in the network. ACWA continues the process (unlike functional analyses typically used in Systems Engineering; see Bahill & Gissing, 2006), and extends the framework of the goal structure to identify the design requirements that must be captured in the HCI. It is because of this extension of the functional analysis that the tighter integration between human and technology is possible.

After the functional analysis identifies the goals that are accomplished, the next step of the ACWA process is the extraction of the cognitive work that must be completed in order to satisfy the goals. This cognitive work encompasses the decisions that the JCS might be asked to make - indicating what the operator must use the HCI for within the JCS. By identifying the work that must be done, system designers will have the ability to ensure, a priori, that the proper requirements for the subsystem have been developed.

In order to accomplish the cognitive work that is required, the decision-making agent of the JCS needs to have the correct information (the information that must be passed between subsystems). Therefore, the information needed to complete the cognitive work is determined next in the ACWA process. Having the proper information allows the operator to have a greater understanding of the events in the world as well as the actions of the other subsystems within the JCS. This allows for better interaction between subsystems, creating a more seamless integration between these subsystems.

This integration is especially important when the operator is asked to step in when automation has reached its limits. Any information needed for decision-making, whether normally performed by the automation or operator, should be available in the HCI. If the operator is fully aware of what is going on, when it is going on, the transition from automation to operator becomes simple. In this way, the extended functional analysis offers a solution to achieve the necessary "bumpless transfer" between automation and manual modes.

Presenting information in a representation design is different than presenting just data on the HCI. With information, operators are able to make decisions. When only data are provided, the operator is forced to create the information "in the head". At best, this transformation is done properly, only adding time for the decision to be made. At worst, the user makes a mistake in the transformation, leading to a poor decision. In either case, forcing the user to make the transformation increases the likelihood of an error by the JCS.

When creating design requirements, then, these information needs (and not just data needs) must be supported. The requirements must specify how the information is to be presented, so that the operators can use this information to complete the goals with which they have been assigned. The link from the initial analysis to the design requirements provides a traceability ensuring the visualizations presented on the HCI allow the operator to satisfy the identified goals. These requirements define the goals and scope of the information representation in terms of the cognitive tasks it is intended to support (Woods, 1991). Further, they serve as an explicit documentation of the intent of the visual presentation, independent of the technologies available.

The design requirements need to define the shaping and processing for how the information/relationships should be represented to operators (Gualtieri, Szymczak, & Elm, 2005). Unlike typical requirements that specify *what* must be included on the screen (using the "system shall" language), design requirements for the HCI must also specify the representational requirements for *how* the information relationships are to be portrayed on the screen. They provide the requirements for the designer to be more explicit about the specific representation dimensions to utilize in a given design concept.

To define good requirements, according to Gualtieri et al. (2005), the system designers must determine what “the frames of reference for the display should be, what events should be displayed, and what contrasts should be made salient.” These decisions will be based largely on the work that needs to be accomplished and what the information relationships specify about completing this work.

The design requirements should provide an explicit link between particular aspects of the display concepts and the specific work they are intended to support. They should lead to visualizations that are explicitly tied to work that needs to be accomplished and the goals that will be satisfied by completing that work. This link back from representation to goal is important in ensuring that the operator can accomplish the system goals through the HCI. Therefore, validation of requirements not only entails that the stated requirements are present within the system design, but also that the requirements support the identified cognitive work (by capturing the information needs of the user). In this way, it makes it easier to validate that the requirements for the HCI are truly correct for the system. The design requirements produced using functional analysis enable validation aimed at determining the effectiveness of the system.

By designing based on requirements which are linked to the work the user needs to accomplish, the resulting HCI elements can be trusted to support the operator. The user will have a good understanding of what is happening in the work domain, and is able to work with the technology in order to make correct decisions. By enabling the operator to effectively absorb the information necessary to make decisions within the domain, the operator and the technology being used become more tightly coupled; the operator is able to control, through the HCI, the tools of the system more effectively. When the HCI is designed properly, the operator becomes unaware of the existence of the HCI. Instead, the operator sees only events in the work domain, as if the HCI is a perfectly transparent window. When this is achieved, true integration of man and technology is achieved, and the two will be able to work cohesively in accomplishing the goals of the system as identified in the functional analysis.

A case study

Here we present a case study of the application of ACWA to the design of an HCI. Gualtieri et al. (2005) describe the process of generating design requirements and design concepts from a functional analysis (ACWA). This can be directly compared to the type of display that is often produced using the standard Systems Engineering process.

The design described by Gualtieri et al. (2005) is only one piece of a large display suite, but it is illustrative of the ACWA process from which an HCI can be designed. The display was designed to support only one subset of the goals for which the JCS was responsible. A key goal for system performance they identified within the domain was the need to manage a set of intelligence requests (ensuring their completion). Satisfying this goal assists in the completion of other goals within the domain. Within the JCS, the operator was tasked with monitoring this process.

The Systems Engineering process would identify this system goal, and understand the need for user monitoring of this process. However, this is where the functional analysis stops for most system designs. Having identified this system goal (and other, unrelated system goals), the designer would then be able to create a set of requirements that would be used to create the HCI. The typical requirements (Table 1) generated in the design process will state what must be provided for on the HCI. They indicate what data (priority, status, criticality, and time due) must be included in order for the operator to successfully complete the goal. From these, the HCI is then developed.

Table 1. A set of system requirements that specify what needs to be included within the HCI in order for a user to be able to satisfy the system goal.

Typical Design Requirements
1. The system shall provide all intelligence requests 2. The system shall allow a user to monitor the status of each intelligence requests 3. The system shall provide the user with the criticality, priority, and due date of each intelligence request

This type of requirements language will generate a display for the HCI that is similar to the one in Figure 3a. The display has the data, specified in the design requirements, which are appropriate for the completion of the stated goal. The operator is able to find each intelligence request and see its status, priority, criticality, and time due. This would technically be a satisfactory display for working in the domain. As an added benefit, the designer would even likely add usability features, include turning status into a dual coded 'stop light indicator' to make it clearer to the user what the status of each request is and providing the ability to sort based on each category available. These features are typical of making a display "user-friendly" and are considered the necessary ingredients for human-system integration.

However, despite having the required data contained within the display (as specified by the design requirements), the proper information relationships are not represented; because of this, true integration has not been achieved. The design is flawed in several ways. One key problem is that the display makes it difficult to account for the entire set of intelligence requests at one time. The user may have difficulty determining which intelligence request should be serviced next or which requests can be ignored in favor of others. It will also be difficult to see which requests will not be able to be serviced, and therefore, should be ignored. While these data may be present within the display, the chosen representation does not present the information in a way that promotes understanding of the work domain for the user.

Things are further complicated due to the fact that the large data set requires that the user scroll to see many of the data. This forces the user to hide data that may be important, and may cause the user to miss important changes in the data. In any system that requires the user to obscure data in some form, the user is placed in a keyhole which does not support the need for operators to redirect their attention to important events in the environment (Woods & Hollnagel, 2006). While the spreadsheet type display that is shown in Figure 1a organizes the data in a factually accurate manner, this organization does little to support the user interaction with the domain. The user is able to find the data, but must work hard to make the data meaningful. The design requirements have failed to provide enough guidance to create a good HCI for the operator.

This design can be contrasted with the representation design produced using the expanded functional analysis. For the stated cognitive work of monitoring the intelligence request process, Gualtieri et al. (2005) identified a set of information relationships needed to support the work. Each of these information relationships was identified during the analysis portion of ACWA as being important to managing the set of Intelligence requests.

Table 2. The table presents examples of information relationships, and the design requirements based on them, that were created for an HCI visualization (taken from Gualtieri et al., 2005). Notice the directives within the language about how the designer should implement information within the screen, and not just what elements of the screen must accomplish.

Information Relationships	ACWA-based Design Requirements
1. Level of daily success 2. Status of each Intelligence Result 3. Priority of each Intelligence Result – the level that a result supports the commander's daily intent 4. Criticality of the result – the level that a decision or a set of decisions is dependent upon the obtainment of a result 5. Time remaining until a result must be complete 6. Time passed since a result was due	1. Depict time daily with frame of reference to current day 2. Spatially separate fulfilled and not fulfilled results 3. Depict each result as an independent symbol of equal screen real estate 4. Create higher salience for high criticality 5. Create higher salience for high priority 6. Create low level of salience for successful results 7. Create medium level of salience for tasked results 8. Create highest level of salience for failed results

From these information relationships, design requirements are then created that capture the intent behind the information relationships (both the information relationships and the design requirements can be seen in Table 2). These requirements define how the representation on the HCI will portray the information relationships on the screen and define the levels of importance of each element to be represented on the screen by defining the relative salience of each item. By examining the set of information relationships, the designer is able to determine the proper frames of reference to be used on the display (in this case, time and fulfillment status are identified as the two main frames of reference – design requirements 1 and 2), what each mark on the display is to mean (an intelligence result – design requirement 3), and how to code the different marks on the display (design requirements 4-8 specify these encodings).

The language used for the ACWA-based design requirements is vastly different than the language used for typical requirements. Each ACWA-based requirement provides specific instructions to the designer about key elements to which the design must adhere (in this way, the requirements don't contradict the set of requirements for standard HCI design listed in Table 1; they simply provide more direction and necessary constraints for design). While some design choices are still left up to the designer, following the requirements will lead to an HCI that supports the cognitive work of the user. Each design requirement can be directly traced back to each part of the analysis that preceded it; the design requirement will not only support the goals of the domain (as requirements do in most systems), but also the cognitive work, and the information needs as well.

Notice the impact that these RDRs have on the representation of the HCI (Figure 1b). Each intelligence request is now placed on a single, common plot (as specified by design requirement 3) – making it easy to manage the entire set of requests. All requests can be seen on the display at the same time, something that a list does not allow once the number of items in the list gets too large. Time until (or since) the request is due is used as the x-axis of the display (as specified by design requirement 1) and request fulfillment is used as the y-axis (as specified by design requirement 2 – the display shows the number of intelligence requests that have been fulfilled or not fulfilled with the x-axis). The remaining design requirements address salience management, which requires an understanding of how people perceive objects in the world to get correct (Elm et al., in press). The stop light indicator, which ignores the importance of salience, is removed. Instead, status is shown through a combination of position and color – the colors chosen reflect the known perceptual abilities of humans. The entire visualization is made so that users can more easily ensure that intelligence requests were satisfied as required to make the system successful.

In the visualization created from ACWA-based design requirements, the operators are no longer looking at data. The data in each display are identical; the display designed with ACWA-based design requirements, however, converts the data to manageable marks on the screen, while the display designed with the typical design requirements presents only the data, forcing the operators to make this conversion themselves. Forcing a "teammate" in the JCS to complete unnecessary work, as in the HCI in Figure 1a, makes for poor integration. Only when the operator has a clear understanding of the work domain through a visual representation on the HCI (as in Figure 1b) is integration achieved.

The case study provided here highlights what kind of HCIs can be developed when the functional analysis is expanded beyond the identification of goals of the system. The ACWA-based design requirements helped to shape a visualization that reduced the cognitive work the operator needed to perform while increasing the understanding the operator has of the work domain. Transitioning from the text-based display (Figure 1a) to a representation-based display (Figure 1b) reduces the amount of effort the operator must put in to understand the data, allowing the operator to understand the work instead (Elm et al., in press). **The transformation to information that can be used by the operator is done in the display instead of "in the operator's head" – making it easier for the operator to satisfy the system's goals**.

Conclusion

Integrating the operator and the system's technology requires careful consideration of the operator's needs. Providing a means to interconnect two systems does not ensure that the two subsystems will be working in a cooperative manner (Yu, Du Bois, Dubois, & Mylopoulos, 1995). **Without a correct set of functional design requirements, the HCI will only connect the operator and the technology, rather than integrate the components within the JCS.**

To form this integrated team of operator and technology (i.e., the JCS), a functional analysis must be expanded beyond the identification of goals; the functional analysis must uncover the cognitive work that is needed to satisfy the network of goals and the information relationships that are needed to complete this work. This process leads to the generation of design requirements that are explicitly tied to the interconnected set of functional goals of the domain, ensuring that the visualizations created from them will allow the operator to effectively satisfy these goals.

Creating requirements for the HCI that capture the information needs of the operator makes the overall, integrated JCS design truly user-centered, going far beyond the usability and functionality of traditional HCI design. While the HCI must successfully implement this traditional usability and functionality, it must also provide the added dimension of reducing the cognitive work the user needs to do, making it easier for the JCS to complete system goals. **Completing the functional analysis, as specified in ACWA, provides an approach that has a deeper impact than on merely the look and feel of the HCI; it specifies design requirements for an HCI that integrate the human and technology into a fully effective JCS.**

Intelligence Requests

File Edit Help

Request Name	Request Status	Criticality	Priority	Requested Result time
Alpha_1	Successful	High	High	19-Sep
Alpha_2	Successful	Low	High	17-Sep
Beta_1	Tasked	High	High	24-Sep
Gamma_1	Tasked	High	High	23-Sep
Echo_1	Untasked	High	High	30-Sep
Echo_2	Failed	High	High	16-Sep
Beta_2	Untasked	Low	High	21-Sep
Beta_3	Tasked	Low	High	21-Sep
Foxtrot_1	Tasked	Low	High	24-Sep
Foxtrot_2	Successful	High	High	18-Sep
Alpha_3	Successful	High	High	17-Sep
Beta_4	Tasked	High	High	25-Sep
Gamma_2	Untasked	Low	High	26-Sep
Gamma_3	Failed	Low	High	18-Sep
Charlie_1	Untasked	High	High	24-Sep
Charlie_2	Untasked	Low	High	28-Sep
Alpha_4	Untasked	High	Low	30-Sep
Gamma_4	Successful	Low	Low	18-Sep
Echo_3	Successful	High	Low	16-Sep
Alpha_5	Successful	High	Low	15-Sep
Beta_5	Successful	Low	Low	19-Sep
Gamma_5	Successful	Low	Low	20-Sep
Charlie_3	Successful	Low	Low	14-Sep
Charlie_4	Untasked	High	Low	23-Sep

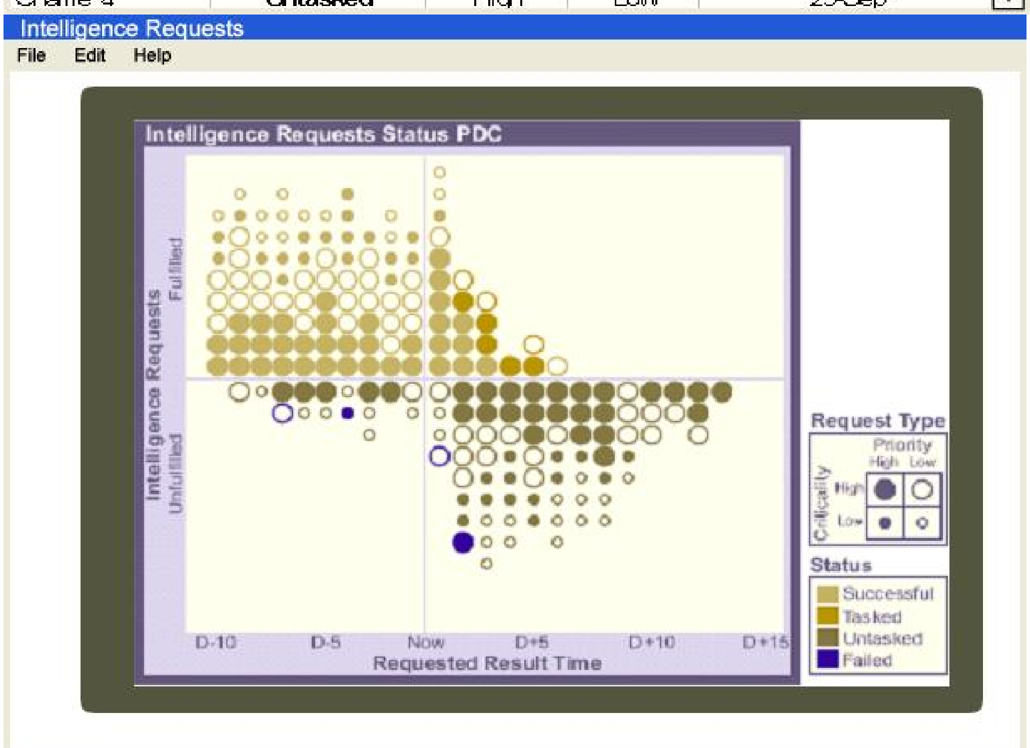

Figure 1. A comparison of two very different displays that were generated to support the same task. A) Based on typical design requirements, this notional display lists each intelligence request and provides the data that was identified to be important for completing the system's goal. B) Using ACWA-based design requirements, the display uses a visual representation to capture all the data needed by the operator in a manner that reduces the cognitive work required to satisfy the goals.

References

Bahill, A.T., & Dean, F.F. (2006). *What is Systems Engineering? A Consensus of Senior Systems Engineers*. Oct 25th, 2006, http://www.sie.arizona.edu/sysengr/whatis.whatis.html.

Bahill, A.T., & Gissing, B. (1998). "Re-evaluating Systems Engineering Concepts Using Systems Thinking". *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews*, 28(4), 516-527.

Deloach, S.A., Wood, M.F., & Sparkman, C.H. (2001). "Multiagent systems engineering". *International Journal of Software Engineering and Knowledge Engineering,* 11(3), 231-258.

Goguen, J.A., & Linde, C. (1993). "Techniques for Requirements Elicitation". *Proceedings of Requirements Engineering*, 152-164.

Elm, W.C., Gualtieri, J.G., Potter, S.S., Tittle, J., & McKenna, B.P. (in press). "Pragmatic Use of CWA in System Design: Extending Current Thinking by Adapting the Mapping Principle".

Gualtieri, J.G., Szymczak, S., & Elm, W.C. (2005). "Cognitive Systems Engineering – Based Design: Alchemy or Engineering". *Proceedings of the Human Factors and Ergonomic Society 49th Annual Meeting.* 254-258.

Hollnagel, E. & Woods, D. D. (2005). *Joint cognitive systems: Foundations of cognitive systems engineering*. Boca Raton, FL: CRC Press / Taylor & Francis.

Hourizi, R. & Johnson, P. (2001). "Unmasking Mode Errors: A New Application of Task Knowledge Principles to the Knowledge Gaps in Cockpit Design." In Hirose, M. (ed.) *Proceedings of Human Computer Interaction: Interact 2001*. IOS Press. 255-262.

McKenna, B.P., Gualtieri, J.G., & Elm, W.C. (2006). "Joint Cognitive Systems: Considering the User and Technology as One System". *16th Annual INCOSE Symposium*.

Woods, D.D., & Hollnagel, E. (2006). *Joint Cognitive Systems: Patterns in Cognitive Systems Engineering.* Boca Raton, FL: CRC Press / Taylor & Francis.

Woods, D.D. (1991). "The Cognitive Engineering of Problem Representations". In G.R.S. Weir and J.L. Alty, (Eds.), *Human-Computer Interaction and Complex Systems*, Academic Press, London.

Yu, E., Du Bois, P., DuBois, E., & Mylopoulos, J. (1995). "From Organization Models to System Requirements – A 'cooperating agents' Approach". *Proceedings of the 3rd International Conference on Cooperative Information System.*, 194-204.

Biographies

Brian McKenna – Brian McKenna graduated with an M.S. in Human Factors Psychology from Wright State University in 2005 and has been working as a Cognitive Systems Engineer since late 2004. Brian worked as a research assistant both as an Undergraduate as well as a Graduate Student covering research areas such as stimulus-response mapping, manual control theory, and Cognitive Systems Engineering.

James Gualtieri - Dr. James Gualtieri has over 15 years experience in applying Cognitive Systems Engineering principles to the analysis of complex work domains and the design and evaluation of decision support systems for both government and private industry. He has conducted research in the areas of system utility, decision making, mental model representation and measurement, and team information processes.

William Elm – Mr. Elm has over 23 years experience in the use of Cognitive Systems Engineering toward effective Joint Cognitive Systems, dating back to CSE's inception with David Woods in the early 80's. He has BSEE and MSEE degrees from Carnegie Mellon University and advanced coursework in Artificial Intelligence at CMU as a Westinghouse B.G. Lamme scholar. Bill started in Cognitive Systems Engineering designing Advanced Control Rooms for commercial nuclear power plants, resulting in several patents, including an alarm management system that still defines the state of the art.