Automating functions in multi-agent control systems: supporting the decision process

M. D. Harrison, P. D. Johnson and P.C. Wright
Department of Computer Science, University of York,
Heslington, York, YO10 5DD. UK
{mdh,pcw}@cs.york.ac.uk

Abstract

A challenge for designers of safety critical systems is to recognise that human issues are critical to their safe automation. Appropriate techniques for taking account of these issues should be integrated into the design process. This paper provides a brief introduction to a two-step decision procedure for deciding how to automate an interactive system. The method has received preliminary evaluation in aviation and naval contexts. It is intended for use early in the development of systems in larger scale collaborative settings with the aim of improving their safety and performance. The method aids the designer in identifying how to take advantage of the benefits offered by automation so that it does not interfere with the operator's ability to perform his or her job. It does this by guiding the matching of functions to a set of defined roles.

1 Introduction

For the automation of complex processes to be achievable with flexibility and safety, the need for procedures[1] that help decide how to automate becomes more important. Work systems are often complex interactive systems involving many people and many technology components. The work that is involved must be implemented in a way that is most compatible with roles that are designed for the people involved. Any automation should satisfy general criteria such as minimising workload, maximising awareness of what is going on, and reducing the number of errors. This paper is concerned with these procedures, known as function allocation, and as such is centrally concerned with the design of safety critical systems.

A number of methods of allocation of function have been developed since the early 1950s. In the main

[1] If they exist, see [Fuld 2000] for the case against existence.

- They fail to recognise fully the context in which the functions are to be automated. Functions are considered in isolation using general capability lists that describe what people are better at and machines are better at. More enlightened methods based around task analyses (for example MUSE, [Lim & Long 1994]) do not give explicit account of the environment in which the functions are performed. A richer description of the context for the analysis needs to be described.
- They are difficult to apply. They depend on an understanding of human factors that requires considerable training. For example the KOMPASS method [Grote et al. 2000] describes criteria for complementary system analysis and design such as process transparency, dynamic coupling, decision authority and flexibility. These criteria are described in terms that are very difficult to interpret by systems engineers.
- Methods do not use design representations that are familiar to systems engineers. Allocation decisions are usually binary. As a result a function is given the attribution "to be automated" or not and consequently must be described at a low level of granularity. Functions are not presented in a form that is easy to assimilate and use by software or systems engineers.
- They focus on the individual relationship between human and device and do not take into account the broader collaborative system and the roles defined therein.
- They offer no guidance about how dynamic allocation of function should be implemented. This problem becomes more relevant as the possibility of dynamic adaptation becomes more realistic.

More recently, methods of function allocation have emerged ([Older et al. 1997] and [Johnson et al. 2001b] for reviews) that take more account of some of these issues. Two methods: referred to herein as the 'Sheffield method', developed at the Institute of Work Psychology, University of Sheffield [Older et al. 1996] and the 'York method', developed at the Department of Computer Science, University of York [Dearden et al. 2000] overcome some of the deficiencies described above but not all. In particular, the Sheffield method is weak on use within a design context and the York method does not address the collaborative issue. What we describe here is a modest extension to the York method aimed at addressing some of the concerns about collaborative issues. Further extensions, not discussed here for lack of space, aim to integrate the inputs and outputs to the method with the notations of UML [Rumbaugh et al. 1999].

If function allocation were to be made an explicit step in the systems engineering development process, it would provide an ideal point to integrate human factors techniques into the development process. The desire for such integration exists because human factors techniques offer solutions to problems faced by engineers in the design of highly automated control systems. Designers of such systems face a dichotomy that can best be illustrated by way of an extreme design philosophy, that of automating everything possible. In a modern control system this could be achieved for a high proportion of the functions and has several advantages. Human operators are a source of error, and by minimising their role these risks can be reduced. Automation also offers the potential for greatly reducing costs. However, Bainbridge [Bainbridge 1987] notes that this design philosophy is flawed because humans cannot be eliminated entirely. When situations arise that the automation cannot handle, the operators are expected to step in and resolve the situation. Their ability is impaired by the high levels of automation that have been introduced, resulting in out-of-the-loop performance decrements and loss of situational awareness. An alternative design philosophy would be to keep the operator in the loop and to view the automation as assisting the operator. The design goal is to take advantage of the benefits offered by automation, but to do so in a way that does not impede the operator's ability to perform his or her role. The achievement of this goal is difficult because it requires the combination of two disciplines, systems engineering and human factors engineering, neither of which alone can provide the solution.

This paper describes one method of function allocation. The method is a modest extension of the York method [Dearden et al. 2000], that in addition provides the means to allocate function to multiple roles, as would be found in a collaborative system. The paper's structure is the following. Section 2 gives a brief description of the method and then describes in more detail what is required as input to the method, and what is produced as a result. Section 3 provides a more detailed description of the method in particular the decision steps that are taken. Section 4 concludes by briefly describing further extensions for which there is no space to elaborate, and discusses further directions.

2 Inputs and outputs to the method

2.1 Introduction

Our method refines a set of function definitions in two steps. A function describes a unit of work that is to be performed by the proposed system. In the first step the functions are matched to a set of predefined roles. These roles capture the high level function of the agents. The aim is to decide how closely these functions fit the roles in a set of given scenarios. Functions that are entirely subsumed within a role are proposed to be "totally manual within the role" – to automate would in effect remove part of the agent's role. Functions that can be separated entirely from any of the roles, and for which it is possible to automate, are designated as "to be automated". In practice few functions fit into either of these categories. In the second step, the functions that are left are considered in more detail in order to decide what aspects of them should be automated. In the following sub-sections we describe function, role and scenario in more detail. Other inputs to the method, for example: description of the technology baseline, mandatory constraints and evaluation criteria are also required as inputs. Function, role and scenario are however the minimum needed to get a reasonable understanding of the method. The output of the method is a set of functions. These functions will either be described as manual or totally automated, or in the case of the partially automated functions, a further description is required of how the function is to be automated.

2.2 Role

Whereas in the early applications of allocation of function, the problem was to decide what people are good at and what machines are good at, recent methods have questioned this assumption. Usually the appropriateness of automation is contextually determined. It makes more sense to be specific about how compatible the tasks are with the various roles specified for the various personnel than to ask the more simplistic question: “would it be sensible for a human to do this?”. In practice role is difficult to define. It makes sense to consider it as an activity that can be performed either by human or machine. Normally it will not be necessary to produce a statement for a machine’s role, but there may be exceptional circumstances where it may be appropriate. For example, the envelope protection provided in a fly-by-wire aircraft could be clarified through an explicit role statement. Doing so would help to ensure that functions are not allocated in a way that prevents the machine from being able to keep the plane within the safety envelope.

An example of a role statement for the captain of a civil transport aircraft is:

> *The captain is the final authority for the operation of the airplane. He must prioritise tasks according to the following hierarchy: safety, passenger comfort and flight efficiency.*

In the context of a single seat military aircraft, an example role statement for the pilot might be:

> *The pilot is the final authority for the use of offensive weapons and for evasive manoeuvres. The pilot is responsible for mission planning, tactical and strategic decision making and co-ordination with other aircraft. The pilot is responsible for the management of systems to maximise the probability of successful mission completion.*

2.3 Functions

Work systems perform functions or *units of work*. Examples of what we mean by function include “finding the current position of the vehicle”, “withdrawing cash” or “detecting a fire”. In order to perform function allocation, a set of such functions is required. Although in practice, identifying and determining a function is largely a matter of expert judgement, some characteristics of a function can be determined that aid this process. A function is an activity that the man-machine system is required to be capable of performing in order to achieve some result in the domain under consideration. Functions do not contain any indication of who or what performs them [Cook & Corbridge 1997]. For example, ‘key in way-point’ is not a function because it implies that the operator enters the way-point manually, whereas ‘set way-point’ does not. Functions can be seen as being related to functional requirements as used in systems engineering. As function allocation is a design technique, a set of functional requirements will be available from which to derive functions. Care has to be taken when doing so because functional requirements often contain indications about who or what should perform the function. Such information needs to be removed.

It is important that functions are defined from a holistic and evenly balanced perspective across the systems components. Otherwise bias will adversely affect decision-making about how functions should be automated. It is therefore necessary to be aware of what viewpoint the functional requirements are taking so that this neutrality can be achieved. Functions can be defined at a variety of levels of granularity, and are often arranged hierarchically. Taking a top-down hierarchical approach can be a useful aid to function elicitation. Descending the hierarchy reveals both a decrease in the complexity of the function and in the size of the sub-system required to perform it. At the top of the hierarchy will be those functions that can only be performed by the system as a whole. Further down will be those functions that are performed by a team or department. Even further down are those functions that can be performed by a single operator with the aid of automation. Finally at the bottom are those functions that can be performed entirely by a single operator or machine. The overall pattern will depend upon the size of the system being modelled.

2.4 Scenarios

Traditionally, function allocation methods presume that the suitability of a function for automation is based on an individual function. We have noted already that this ignores the complex interactions and dependencies between activities of work. To overcome this, the method considers the allocation of functions together as groups within the context of a scenario. This allows the designer to appreciate the interactions between the functions. Scenarios should be selected that focus upon situations relevant to critical decision criteria, for example workload or situation awareness. They should represent all the functions that have been selected in a range of contexts. For example, if the system is in the field of civil aviation and workload has been identified as a decision criterion then scenarios focusing upon take-off and landing would be chosen (among others such as emergency conditions) because these periods are recognised as having a high workload.

As the method only considers those functions used within the current scenario, the designer must ensure that every function occurs in at least a minimum of one scenario and ideally several. It is important to ensure that a wide variety of scenarios are used so that confidence can be placed in the results. In order to achieve this, scenarios should be selected that cover all the normal operating conditions of the system.

There is a wide range of sources for possible candidate scenarios:

- The experience of practitioners in previous systems
- Incident and accident reports for previous systems
- Scenarios developed during the business modelling stage of the system life cycle
- Use cases and scenarios in the previous systems development documentation or training manuals

When a scenario is adopted, the description of what happens is likely to be at an event level. For example, if the scenario is describing some current possibility with the baseline architecture, these events will be expressed in implementation dependent terms. It is therefore necessary to transform these scenarios into a form in which the account of what happens is translated into functional terms. The functions being unbiased towards the implementation, will be less likely to be tied to the assumptions made in the baseline. It must be noted however that a general problem with this sort of approach is that there will be some bias.

It is unlikely that all of the candidate scenarios will be useful for the purposes of function allocation, or that they will be in the correct format. The designer therefore will have to select those appropriate and document them. Checking that they are appropriate will involve transforming each of them into a format based on functions rather than events. The scenario representation is based on a modified version of the scenario template for THEA [Pocock et al. 2001]. The template used for the scenario descriptions is shown in Figure 1.

UC#	The name of the use-case	
Scenario 1 - Each use-case can have a number of scenarios		
Environ-ment	A description of the environment within which the system is operating when the scenario occurs.	
Situation	A description of the state of the system at the start of the scenario. Are all the operators on duty, are there any known or unknown faults in the system etc.	
Sequence of events	**Step**	**Event**
	1	The main sequence of events that take place during the scenario. This includes events that happen in the environment, events that effect the system and the actions that the system must perform.
	…	
Event extensions	**Step**	**Event**
	1	Variations upon the main sequence of events are recorded as event extensions.
	…	
Scenario 2		

Figure 1 Scenario input template

The scenario as illustrated in Figure 1 is taken as a starting point. This description is then transformed. The events and event extensions headings are changed to functions and function extensions. The account of what happens in the scenario is re-expressed using the set of functions rather than the event descriptions. This process can be a useful check that the functions are expressed at an appropriate level and whether they incorporate too much implementation bias. The functional scenarios are then used as the basis for the two decision procedures.

2.5 The nature of the output

Traditionally function allocation methods provide two allocation options H (human) or M (machine). It is assumed therefore that functions are at a very low level so that such binary decisions can be made. In many cases the designer’s interest lies with those higher level functions that require some form of collaboration between operator and machine. It is the determination of where this automation boundary lies that is of critical importance. Rather than working with numerous low-level functions to determine this boundary, it is easier to work with the parent functions and to declare them as partially automated. Of course there are many ways in which the operator and machine can interact in the fulfilment of a function. There is therefore a need for the designer to specify how the partnership will operate.

One possible approach is to use a classification that defines levels of automation. [Sheridan & Verplanck 1978] suggest a classification of levels in which decisions and control pass progressively from the human to the machine. Later authors have produced alternatives. Kaber and Endsley provide ten levels of automation [Kaber & Endsley 1997]. Billings suggests seven levels of management automation [Billings 1991]. For each function that the designer defined as being partially automated, there would be an indication of what the level of automation would be. The difficulty with this approach is that defining which level of automation is required does not necessarily define what it means for the particular function to be implemented. Another problem with such an approach is that some solutions for functions may not fit into any of the classifications.

The approach we take is to provide a conceptual framework for thinking about partial automation. The framework makes use of the [Malinkowski et al. 1992] framework for adaptive systems and is capable of expressing all types of automation provided by the various classifications. A function is first split into four components: Information, Decision, Action and Supervision. Each component is further split into a number of elements, each describing a particular aspect of the function. The designer specifies which role is responsible for performing that aspect of the function. If the element is not applicable within the context of the function then it is marked not applicable (n/a).

The elements in the Information component cover issues such as which role integrates the information required to carry out the function, and which role is responsible for initiating a response. The Decision component covers such issues as which role proposes what plan/action to take, evaluates it, modifies it and selects one, if there is more than one possibility. The Action component covers which role carries out the action. The Supervision component covers such issues as which role monitors the performance of the action, identifies exceptions, and revokes the action if necessary.

This framework (called IDA-S, see [Dearden 2001]) allows the designer to express precisely how the function is implemented in terms of the various roles that are responsible aspects of the function. This is done by placing a role identifier in an appropriate cell in the IDA-S template. In the case of the function: “calculate point to point information”, one possible solution might be that the navigator selects two points upon the chart, that is the current fix and next way-point. The navigation system calculates the distance, time and bearing to the way-point. In Figure 2, the required roles are placed in the appropriate cell of the grid. In some cases, where more than one role appears to be appropriate, they may all be placed in that cell. A complication, such as this, may suggest that the function has not been sufficiently refined at the stage of selection, and may suggest a need to revisit the function

Allocation	**Information**		**Decision**		**Action**	
Planning the response	Collect		Propose		Approve	
	Integrate		Evaluate			
	Configure		Modify			
	Initiate response		Select			
Supervise ongoing execution	Monitor progress		Identify exceptions		Revoke authority	
Supervise termination	Determine output content		Identify completion		Stop process	
Action	Execute actions					

Figure 2: The IDA-S template

definitions and the scenarios. The IDA-S definition clarifies how to develop an implementation that satisfies the requirements.

3 The method: two steps and consolidation

3.1 Totally automated or wholly manual?

Once roles, functions, scenarios, and so on, have been defined the first step involves deciding which functions can be totally allocated to one of the roles (these roles may be system or human). Each scenario is considered in turn and the functions, that are employed within the scenario, are identified. The designer bases decisions about automation of these functions on their use in the context of the scenario under consideration. Suitability for total automation is not based solely upon the technical feasibility of a solution. It is also based upon the function's relation to the roles. If a function is not seen to be separable from an operator's role then it cannot be totally automated. Doing so would interfere with the operator's ability to do the job effectively. There are two dimensions to the trade-off.

Firstly, each function is considered in relation to the feasibility of automating it. The concern here is with the cost and technical possibility. Secondly, the function is matched against the set of roles. The roles are likely to continue to be refined as the design evolves. In Figure 3, two roles R1 and R3 are relevant to the given scenario. In practice, a function may be separable from all roles, and technically feasible and cost effective to automate, in which case the function may be totally automated. Alternatively it is possible that the function maps entirely to one of the roles, and is infeasible to automate, in which case the function is totally performed within that role. In most cases however functions fit into neither category. In this situation the function is to be “partially automated”. There are usually a number of ways in which partial automation may be achieved, and this is the subject of the second trade-off. The first decision procedure is made traceable through the matrix in Figure 3. Notice that a function may appear in more than one row because it relates to several roles, but may only appear in one column because feasibility to automate is invariant.

State of automation research vs. relation to role	Role	Existing with immediate access	Existing in competitor systems	Low risk/low cost R&D	High risk or high cost R&D	Infeasible
Separable	ALL		Sol1			
Role related information or control	R1					
	R3					
Role critical information or control	R1					
	R3					
Central to role	R1					
	R3					

Figure 3. The first trade-off

When this process has been completed, a set of functions has been produced that can either be totally automated or are entirely subsumed within a role. These functions go no further in the process. The remaining functions go forward to the next stage of the allocation of function process. These functions are represented using the IDA-S framework defined above.

3.2 Candidates for the second step

The second decision procedure is concerned with comparing alternatives with a “baseline” solution in the context of a scenario. Often the baseline is the current design, but it may be that, in the case of a completely new concept, there may be a more conservative design possibility. It is assumed here that an initial process is engaged in where the designer constructs a number of alternatives for each function listed from the scenario. These alternatives may allocate different roles to elements of the IDA-S template. For example, in the case of “calculate point to point information”, we could consider one solution in which the navigation system might propose alternative routes from which the navigator selects the most appropriate, while in another no such choice is provided. The reason for choosing particular representations may be random or based on some assumptions about the abilities of the off-the-shelf technologies available to the project. The options for all these functions are then rated in relation to a criterion such as workload or performance in comparison with the “baseline” design.

When all the options for all the functions have been produced, and the primary concern identified, the options are entered into a second matrix (see Figure 4). The design options used in the baseline should also be included in the cell that is labelled “no significant improvement in primary concern”. The alternative solutions are then placed in the table. Two criteria decide where the solution should fit. The first depends again on the feasibility of the particular solution, how easy will it be to implement with existing technology. The second requires a judgement about the effect of the solution, in the context of the scenario, in terms of the criteria (workload, performance, situation awareness). The judgement here is whether the function solution causes an improvement or deterioration to the primary criterion and the consequent effect on the other criteria. All these judgements are carried out in relation to the baseline design. It would be expected that some solutions will do better while others will do worse. It is intended that the designer uses expert judgement, but it could be that the situation requires a more careful human factors analysis of these decisions.

Each potential solution is placed in the matrix. At the end of the process it will be possible to derive a set of best candidates that are the solutions that are most favourable in relation to the criteria and are technically feasible. This process is achieved by searching from the top left of the matrix selecting new design options. If a design option for a function is selected, then all other options for that function are deleted from the table. If a design option is selected from the ‘high risk research and development’ column, then an alternative, low risk solution should also be considered as a ‘fall-back’ position. Hence in Figure 4 two options are provided for a function F.1.2.4. Both are implementable because they are available on competitor systems. However the second solution is preferable because while it results in an improvement in performance compared with the baseline solution, it has no deleterious effect on any of the secondary concerns such as workload.

After a number of options for functions have been selected, the designers should re-evaluate the scenario and consider whether or not the primary concern should be changed as a result of the decisions made so far. For instance, consider a scenario in which high workload is the primary concern. If new partially automated solutions are selected, that significantly reduce the expected workload, then a different concern such as performance or situation awareness may now be more significant. If the primary concern is changed, then options for the remaining functions are re-arranged in a new matrix reflecting the changed priorities. Option selection then proceeds as before, starting with the options that provide the greatest improvement for the new primary concern. This procedure iterates until one design option has been selected for every function.

One possible outcome of the procedure is that some functions cannot be successfully allocated, without making use of options from the high risk research and development column, or from a row involving a ‘large deterioration’ with respect to a secondary concern within the scenario. If this occurs frequently, and cannot be solved by generating alternative design options, this may indicate a need to review the system requirements, or to review assumptions about the number and role of human operators in the system.

When a design option for each function has been selected, the scenario is re-analysed using the proposed allocations as the set of baseline designs. The purpose of this re-analysis is to identify any new functions that may be an emergent consequence of the new design. Such functions could include, for example, a requirement to co-ordinate two separate functions that control the same system resources (e.g. ‘terrain following’ and ‘missile evasion’ both use the aircraft’s control surfaces). Design proposals for the partial automation of these functions are made.

Primary concern:	Performance			
State of automation vs level of improvement	Suggestion is immediately available	Available on competitor systems	Low risk, low cost R&D	High risk or high cost R&D
Large improvement in primary concern, no deterioration in secondary concerns				
Improvement in primary concern, no deterioration in secondary concerns		(6) F1.2.4 Sol 2		
Improvement in primary concern minor deterioration in secondary concerns		(10) F1.2.4 Sol 1		
Improvement in primary concern, large deterioration in secondary concerns				
No significant improvement in primary concern				
Large deterioration in primary concern				

Figure 4: Identify partially automated functions

If new functions are identified, then designers must consider whether their impact upon performance, workload and situation awareness is acceptable. If the emergent functions do create an unacceptable situation, then the selection matrix is revisited. This may result in changing the level of automation, or may result in changed selections for the original functions. Hence, if emergent functions are identified then steps of the process dealing with these are repeated, i.e. feasible design options are suggested for partially automating the emergent functions and the optimum choice is selected.

3.3 Consolidation

Once the functions within each scenario have been allocated any contradictions of allocation across scenarios are resolved. This is done, either by changing one of the allocation decisions so as to resolve the conflict, or by allowing the function allocation to be dynamic. Allocation conflicts may be resolved by choosing solutions that are most favourable *across the scenarios* using the primary and secondary criteria of the second trade-off. There may be situations where this global compromise is difficult, if not impossible, to make and this may occur in circumstances where the level of automation does not have to remain fixed. Components of IDA-S allocations can be transferred from one role to another during the activity supported by the system. Dynamic function allocation refers to the process of redistributing functions amongst roles, with the goal of improving overall system performance. This redistribution typically occurs in response to a change in the environment or a change in the state of one of the agents assuming the role. The designer must decide how the allocation of function changes, and the extent to which the operator is involved in this process. In practice, the change-over can be seen as another function to be refined, in the same way as any other function that emerges during the process.

When a function is redistributed amongst human agents it is often referred to as workload sharing. When a function is redistributed between human and machine it is often referred to as adaptive automation. Redistribution can either be human-triggered or machine-triggered. Much current research focuses on dynamic allocation in response to changes in workload. Triggers for reallocation also include, for example:

- Critical system events – adaptive automation is triggered if a system error occurs
- Performance measurement - degradations in human vigilance invoke adaptive automation
- Psychophysical assessment - dynamic assessments of human physiology drive the decision as to whether to automate
- Behaviour modelling - control allocations are used to achieve a predetermined pattern of overall system functioning [Parasuraman & Mouloua 1996].

A detailed description of this stage of the process can be found in [Johnson et al. 2001a].

4 Extensions and conclusions

There has not been space to discuss the work that has been done to integrate the method with notations and representations that are relevant and comprehensible to systems engineers. The main steps here have been to develop an extension to the use-case mechanism of UML [Rumbaugh et al. 1999] in order to provide a more readily accessible representation of scenarios, and to use activity diagrams from UML to describe IDA-S patterns. The difference between the templates described in this paper and activity diagrams is that a further commitment needs to be made about the sequence in which information, decision, action and supervision takes place which is not contained in the IDA-S template. The UML proposal therefore gives further guidance to the systems engineer about how to implement the functions. More information about this is contained in the fuller paper [Johnson et al. 2001a].

An important issue has been to keep the method sufficiently procedural and straightforward that it can be used in practice. For this reason we are reviewing the complexity of the IDA-S template. We are looking at alternatives that provide attributes that are simpler to understand. An area where the method is perhaps too simplistic and untried is the area of adaptive automation. It may be possible to do better than assume (1) that automation adapts by function substitution and (2) that in all cases the ideal result is achieved. In practice both assumptions may be false. As noted by [Sperandio 1978], in discussions of air traffic control, different strategies are adopted depending on the number of aircraft in the air space. Sequences of functions making up procedures rather than individual functions are substituted. It may also be appropriate that dynamic mechanisms should be prepared to shed certain less critical functions in the face of hard deadlines. Both these issues are discussed in a little more detail in [Harrison & Bernat 2001].

Acknowledgements: We would like to acknowledge the contributions of Andrew Dearden, Colin Corbridge and Kate Cook to this work and thank Shamus Smith for reading draft versions. It has been funded successively by BAE SYSTEMS and DERA CHS.

5 References

[Bainbridge 1987] Bainbridge, L. Ironies of automation. In Rasmussen, J, Duncan, J. & Leplat, J. (eds). New Technology and Human Error, pp 276-283, John Wiley & Sons, 1987.

[Billings 1991] Billings, C.E. Human-Centered Aircraft Automation. Technical report number 103885. NASA AMES Research Center, 1991.

[Cook & Corbridge 1997] Cook, C.A. & Corbridge, C. Tasks or functions: what are we allocating? In E. Fallon, L. Bannon & J. McCarthy (eds.) ALLFN'97 Revisiting the Allocation of Function Issue: New Perspectives, pp. 115-124. Louisville KY: IEA Press, 1997.

[Dearden et al. 2000] Dearden, A., Harrison, M.D. & Wright, P.C. Allocation of function: scenarios, context and the economics of effort, International Journal of Human-Computer Studies 52(2) 289-318, 2000.

[Dearden 2001] Dearden, A.M. IDA-S: a conceptual framework for partial automation. To appear in: Blandford, A., Vanderdonckt, J. and Gray, P. (eds.) People & Computers XV – Interaction without Frontiers. Proceedings of IHM-HCI 2001. Springer. Berlin, 2001.

[Fuld 2000] Fuld, R.B. The fiction of function allocated, revisited. International Journal of Human-Computer Studies. 52(2) 217-233, 2000.

[Grote et al. 2000] Grote, G., Ryser, C., Wafler, T., Windischer, A. & Weik, S. KOMPASS: a method for complementary function allocation in automated work systems. International Journal of Human Computer-Studies. 52(2) 267-288, 2000.

[Harrison & Bernat 2001] Harrison, M.D. & Bernat, G. What can dynamic function allocation learn from flexible scheduling. University of York. DIRC working paper. 2001. www.cs.york.ac.uk/~mdh/papers/harrisonb01.pdf.

[Johnson et al. 2001a] Johnson, P.D., Harrison, M.D. & Wright, P.C. An enhanced function method. University of York www.cs.york.ac.uk/~mdh/papers/johnsonhw01.pdf.

[Johnson et al. 2001b] Johnson, P.D., Harrison, M.D. & Wright, P.C. An evaluation of two methods of function allocation. People in Control IEE Press. Conference Publication No. 481. 178-183. 2001.

[Kaber & Endsley 1997] Kaber, D.B. & Endsley, M.R. The combined effect of level of automation and adaptive automation on human performance with complex, dynamic control systems. In Proceedings of the Human Factors and Ergonomics Society 41st Annual Meeting. pp 205-209. Santa Monica CA.: Human Factors and Ergonomics Society, 1997.

[Lim & Long 1994] Lim, K.Y. & Long, J.B. The MUSE Method for usability engineering. Cambridge University Press. 1994.

[Malinkowski et al. 1992] Malinkowski, U., Kuhme, D.H., Schneider-Hufschmidt, M. A taxonomy of adaptive user interfaces. In Monk, A., Diaper, D. & Harrison, M.D. eds. People and Computers VII, Proceedings of HCI'92. pp 391-414. Cambridge University Press, 1992.

[Older et al. 1996] Older, M., Clegg, C & Waterson, P. Report on the revised method of function allocation and its preliminary evaluation. Institute of Work Psychology, University of Sheffield, 1996.

[Older et al. 1997] Older, M.T., Waterson, P.E. & Clegg, C.W. A critical assessment of task allocation methods and their applicability. Ergonomics, 40: 151-171, 1997.

[Parasuraman & Mouloua 1996] Parasuraman, R. & Mouloua, M. (eds). Automation and human performance: theory and applications. Lawrence Erlbaum, 1996.

[Pocock et al. 2001] Pocock, S., Harrison, M.D., Wright, P.C. & Johnson, P.D. THEA: a technique for human error assessment early in design. IFIP TC 13 International Conference on Human-Computer Interaction. IOS Press, 2001.

[Rumbaugh et al. 1999] Rumbaugh, J., Jacobson, I. & Booch, G. The unified modelling language reference manual. Addison Wesley. 1999.

[Sheridan & Verplanck 1978] Sheridan, T.B. and Verplanck, W.L. Human and computer control of undersea teleoperators. Technical report. Man-machine systems lab, Dept of Mechanical Engineering, MIT, Cambridge, MA, 1978.

[Sperandio 1978] Sperandio, J.-C. The regulation of working methods as a function of workload among air traffic controllers. Ergonomics 21:195-202, 1978.