Proposing a Meta-Language for Specifying Presentation Complexity in order to Support System Situation Awareness

Christoph Endres
German Research Center for Artificial Intelligence (DFKI)
Germany
Christoph.Endres@dfki.de

Michael Feld
German Research Center for Artificial Intelligence (DFKI)
Germany
Michael.Feld@dfki.de

Christian Müller
DFKI, EIT ICT Labs
Intelligent Mobility and Transportation Systems
Christian.Mueller@ictlabs.eu

ABSTRACT

Situation Awareness is an established area of research. In our work, we go one step further and define system situation awareness and identify a way to implement it. One of the prerequisites we identified is the annotation of the complexity of presented information. In this paper, we describe our approach and formulate a proposal for a presentation meta-language aimed at annotating cognitive complexity.

1. INTRODUCTION

Situation awareness, according to Endsley [9, 10] means "knowing what is going on around you". Although Endsley's definition primarily aims at the human operators understanding of its surrounding, we can extend it to the system as well: One of our contributions is the introduction of "System Situation Awareness", which is based on Endsley's model.[6, 3].
As the aim is to minimize the distraction and thus to increase the performance and safety of the driver, the necessary parts of the situation for the system to be aware of are twofold:
1. What is the current cognitive load of the driver? Which cognitive capacities are available for him to process the next incoming information?
2. How complex respectively cognitively demanding is the information to be presented to the driver? How can it be modified to be either more detailed or to easier to understand (in the terms of the previous question)?
Figure 1 shows the two lines of research leading up to achieve situation awareness.

2. SYSTEM SITUATION AWARENESS

We introduce the concept of *system situation awareness*, as the awareness of the system of its environment or rather the extend to which the environment and the user are represented in and considered by the system.
Figure 2 shows an adapted version of Endsley's model. The three levels of situation awareness (perception, comprehension, projection) are replaced by the three levels of system situation awareness.
Level 1, Assessment of user and context, in our case covers cognitive load assessment and presentation complexity estimation. The acquisition of data is the systems equivalent to the human perception.
Level 2 includes updating, storing, and annotating information obtained in level 1, in analogy to human comprehension.
Level 3, comprehension in the original model, is replaced by impact estimation.
In consequence, we see that the human decision making actually is replaced by the concept of intelligent mediation [16]. Instead of performance of action, the presentation of the selected information takes place and influences the state

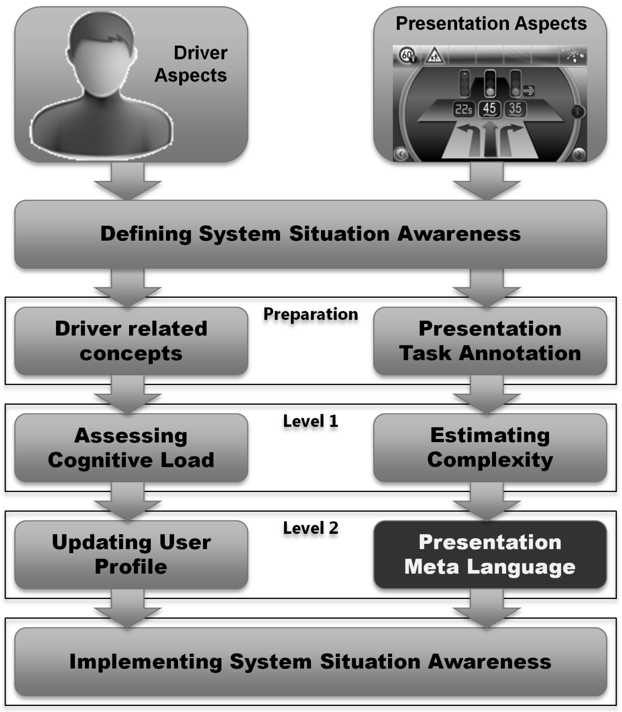

Figure 1: In order to achieve situation-aware presentations, both cognitive load of the driver and presentation complexity have to be considered.

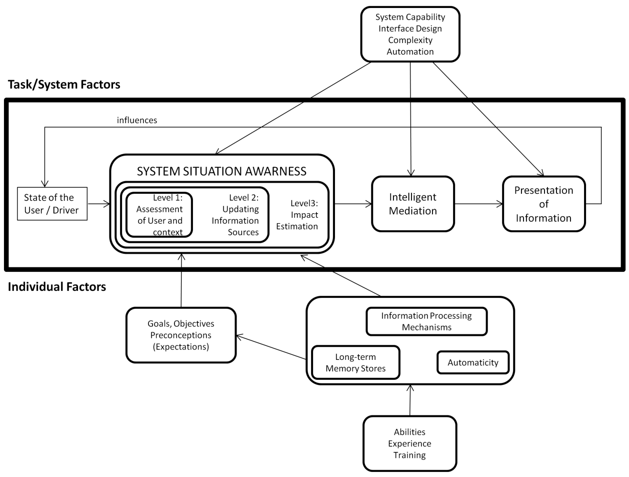
System Capability
Interface Design
Complexity
Automation
Task/System Factors
influences
SYSTEM SITUATION AWARNESS
State of the User / Driver
Level 1: Assessment of User and context
Level 2: Updating Information Sources
Level3: Impact Estimation
Intelligent Mediation
Presentation of Information
Individual Factors
Goals, Objectives Preconceptions (Expectations)
Information Processing Mechanisms
Long-term Memory Stores
Automaticity
Abilities
Experience
Training

Figure 2: We propose the model of System Situation Awareness depicted here as a special case of Endsley's model of Situation Awareness.

of the user, and closes the circle as this is the state to be assessed by the system situation awareness.
The outer parts of the Endsley diagram stay mostly unchanged, except for removing the connection from the individual factors to intelligent mediation and presentation of information, as these now are only indirect connections.

It is important to keep in mind that whenever we reduce a complex concept such as the mental state of a person or the effort required to steer a car to a single number, we lose qualitative information. In consequence, formalizing concepts has to be approached with care in order to obtain a model resembling reality as close as possible.
Cognitive Capacity now refers to another potentially unknown variable, e.g. at what point of the scale the 100 % load limit is reached. Due to individual differences, the cognitive capacity varies among persons and different factors to be discussed later. Another important factor is the ability of human beings to adapt theis performance level in times of higher demands.

3. ELEMENTS OF THE DRIVING TASK

Vehicular guidance in essence is determined by visual tracking and motor/haptic reaction. Most of the driving time, maneuvering is performed, usually in lane keeping or lane changing, but also in turning or parking. The percentage of adjusting speed is depending on the driving environment. It occurs less on a quiet rural road than in busy urban traffic. Reaction to obstacles also is a part of maneuvering.
Navigation can achieve different levels of difficulty. In familiar areas, it becomes an automatism and does not require much resources. In unfamiliar areas however, the complexity of navigation is depending on the tools at hand, ranging from remembering a verbal description over written instructions and paper map up to a navigation system. [17] states that the glance time at a navigation display is significantly longer than glancing on a printed map, which can be attributed also to the lower quality of navigation systems 20 years ago and the driver being unaccustomed to such a device [1].
Communication and social environment become an increasingly important factor in the driving task. While previously limited to conversations with co-driver and other passengers or the car's entertainment system, over the last few years the use of mobile phones, text-messaging and social networks have become a serious issue [7].
Operating and monitoring systems is the fourth component of the driving task. In both cases, critical and uncritical systems are distinguished.
Finally, command decisions are also a part of the driving task, although in comparison less than for a military aircraft pilot. Examples here are passing maneuver, the decision to perform a U-turn, or selection a route depending on the current traffic density or the time of the day.

We define driver distraction as source of diversion of one or more cognitive processing resource away from the primary task with the effect of diminishing the drivers driving performance. Or, in short, distraction is a cognitive demand which is not caused by the requirements of the primary task. Hence, we can classify distractions by the processing resources required, e.g., diminished.
As an example: Loud music from the car stereo would be a distraction in the dimension <audio, driver, inside, intended>. A traffic jam belongs to <obstacle, event, outside, unintended>. A crying baby on the passenger seat is in the category <audio, person, inside, unintended>. Similar, a ringing cell phone is categorized as <audio, person, inside, unintended>.
Based on this classification, we can assign numerical levels to each distraction category and determine the total level of distraction as the sum of all occurring distractions in the implementation of PresTK. The concepts *stress* and *strain* are originally used in the context of evaluating the quality of physical material provide a good metaphor for describing the individual differences of the impact on cognitive demand on the individual cognitive capacity: The same level of stress does not result in the same level of strain for all drivers.

4. AUTOMOTIVE ONTOLOGY

[11] claim that the Human-Machine Interface (HMI) in vehicles is currently undergoing an evolutionary change: "While the focus is still on functions supporting the primary task, a new generation of more powerful interaction, service and entertainment concepts, which extend beyond driving and also involve non-driver passengers, is starting to surface." These developments lead to a greater degree of context-adaptability and personilazability. To support this development, a new and open approach for sharing, maintaining, and exchanging information is required: A knowledge base to assemble important information on environment, vehicle, driver, passengers, and context. Our Knowledge And Personalization component KAPcom is our approach to realize this infrastructure.
Additional to this component, we also need a way to formalize this knowledge.
The KAPcom Automotive Ontology [11] was designed with the following benefits in mind:
1) Functions can easily be made context-adaptive, e.g. dependent on vehicle speed, traffic conditions or surroundings;
2) Applications can exchange knowledge and cooperate in new ways;
3) User models can be shared between vehicles (e.g. car and motorcycle);
4) Privacy features allow a fine-grained control over what is

shared over a car-2-car channel.
KAPcom uses generic user properties and characteristics from [12] and supports time-based reasoning based on the methods described in [15] and thus supports dynamic change of information and keeping a history of previous values/states.

The extendable Automotive Ontology proposed in [11] did not yet contain any means for annotating the cognitive load of a person in detail. In this section we propose a way to extend it and include this information. As we have seen previously, cognitive load can be modeled on different levels of complexity. We can use a single value to annotate the overall complexity, or we can get in more detail into the different processing resources required. Listing 1 shows a simple annotation.

Listing 1: Simple annotation of Cognitive Load in the KAPcom Automotive Ontology

```
<kapcom>
  <user id="m.feld">
    <cognitiveload>
      <overall value="0.3" />
    </cognitiveload>
    [...]
  </user>
</kapcom>
```

To model the current Cognitive Load of the user in more detail, we get back to the processing resources identified previously. Listing 2 shows an example.

Listing 2: Simple annotation of Cognitive Load in the KAPcom Automotive Ontology

```
<kapcom>
  <user id="m.feld">
    <cognitiveload>
      <overall value="0.3" method="mean" />
      <resources>
        <resource name="visual"     value = "0.2">
        <resource name="auditory"   value = "0.3">
        <resource name="motoric"    value = "0.4">
        <resource name="deciding"   value = "0.3">
        <resource name="recall"     value = "0.1">
        <resource name="reacting"   value = "0.5">
        <resource name="spatial"    value = "0.4">
        <resource name="linguistic" value = "0.2">
      </resources>
    </cognitiveload>
    [...]
  </user>
</kapcom>
```

5. ANNOTATED COMPLEXITY ESTIMATION (ACE)

In order to assess system-generated CL, we need to be aware of the impact of system-generated presentations to the driver, i.e., estimate presentation complexity. Determining the complexity value is depending on the availability of structured data. We distinguish three different cases:
1. Structured information about possible presentations is available as a blueprint in the system, or can be derived beforehand from unstructured presentation information.
2. The presentation is available at runtime and specified in a formal presentation markup language.
3. We obtain an unstructured presentation in form of an image or an audio file, or both.
In this section, all three cases are discussed shortly. However, I emphasize on case 2 in the discussion and in the implementation of the PresTKsystem.
Case 1 does not provide much of a scientific challenge and is of little interest here. It merely reduces the question to an issue of solid design.
Case 3 is very hard to tackle at a level any better than by an educated guess. However, it is not completely impossible to reverse engineer the raw data received for presentation and obtain structured data to be subjected to analysis that we perform on structured data. [4] for instance used reverse engineering to transform existing GUIs into a generic description in the language XIML, which could in turn be used for rendering the same GUI on different platforms. The authors did however have the advantage of having a GUI to analyze, and not just a picture to be presented. Analyzing a picture of a presentation to obtain its structure is an interesting challenge, but out of the scope of this thesis.

Case 2 is our focus of attention. Structured data for alternative presentation strategies of the same content can be encoded in the presentation container language defined in section **??**. Here, I introduce the *Annotated Complexity Estimation* procedure *ACE* [8] to formally analyze the complexity. Using this measure, we can provide a value for the cognitive complexity of a presentation at runtime.
A literature review indicates clearly that considerable work has been put into analyzing parameters and conditions to streamline and improve the delivery of information to the driver of a vehicle. Especially [14] and [13] probed every conceivable parameter of in-car display design very thoroughly and provided display designers with a detailed model of their impact to the drivers perception. Attempting to extend their work would not be of much avail.
On the other hand, their work is based on displays of the late eighties, and technological progress did not stop there. While the emphasis back then was on font size, color, brightness, contrast and word complexity, we now also have to deal with sophisticated layouts, background patterns, icons, etc. Also, the use of a touch screen and virtual buttons on the screen was not considered then.

Layout is commonly defined as the part of graphic design that deals in the arrangement and style treatment of elements. We distinguish between grid-layout and the more rigid template-layout. A programming interface for a user interface, such as for instance Java Swing [5], provides several layout managers for the developer to chose from.
In defining the Annotated Complexity Estimation procedure ACE, we reverse the top down layout manager process to a bottom up aggregating model of complexities. Figure **??** shows a sample screen from the sim^{TD} system, and figure 3 provides a schematic view on it.
The nested structure of a user interface can be represented as a tree structure. The main layout manager is located at the root of the tree. Other layout managers may be nested in it.
When analyzing the complexity of a layout, we start at the leaves of that tree and work up to the top and accumulate the complexity until we reach the top of that tree.
The simplest leaf (or more precise: component) we encounter is for instance a label or an icon. A label has a text of a certain complexity, and it might contain an additional small icon making it visibly more complex. These rudimentary components can be grouped in a panel with identical or different elements. The panel has featured like the size in terms of the number of elements it contains, or it might have a visual boundary, such as a borderline, that makes it easier to perceive as a unit.
Panels again might be combined to a higher level panel. Fol-

component	basic complexity	feature	added
label	0.1	text=true	+0.5
		icon=true	+0.4
icon	0.1	type=empty	+0.0
		type=icon	+0.5
		type=static	+0.2
		metainfo=text	+0.4
panel	$0 + \sum$ child nodes	decoration=framed	+0.2
		decoration=none	+0.5
		metainfo=named	+0.2
		metainfo=none	+0.5

Table 1: Calculating values for ACE evaluation

lowing this combining of elements further, we reach the root of the tree and the component that fills the whole screen. We are interested in a numerical value describing the visual complexity of that root note. In figure 4, the structure of

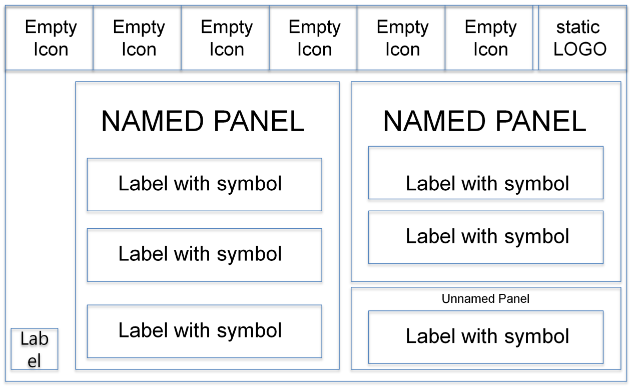

Figure 3: The structure of the HMI as nested GUI components.

the HMI is transformed into a component tree. In order to apply the ACE procedure, the leaves of the tree have to be annotated with numerical values, and for all non-leaf nodes an aggregation formula has to be specified. In order to automate the procedure, we transform the tree into a machine-readable XML annotation. Listing 3 shows the XML representation of the tree.

Listing 3: The structure of the sim^{TD} option screen in XML representation.

```
<ace name="simtd␣options␣screen">
 <root>
  <panel type="icon">
   <icon type="empty" />
   <icon type="empty" />
   <icon type="empty" />
   <icon type="empty" />
   <icon type="empty" />
   <icon type="empty" />
   <icon type="static" />
  </panel>
  <panel type="label" metainfo="named" decoration="framed">
   <label text="true" icon="true" />
   <label text="true" icon="true" />
   <label text="true" icon="true" />
  </panel>
  <panel type="label" metainfo="named" decoration="framed">
   <label text="true" icon="true" />
   <label text="true" icon="true" />
  </panel>
  <panel type="label" metainfo="none" decoration="framed">
   <label text="true" icon="true" />
  </panel>
  <label text="true" icon="true" />
 </root>
</ace>
```

An interim consent was achieved as shown in Table 1. It will be the objective of further experiments to determine its validity or learn more accurate projections.

The algorithm traverses the XML description, recursively

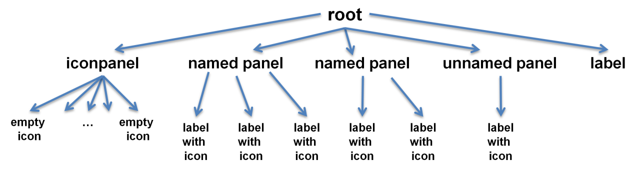

Figure 4: The HMI as component structure tree.

assigning values to each component. The attribute cpx is used for annotation. In our example, this results in annotated XML shown in listing 4.

The overall complexity calculated for this example is 9.3.

Please not that this is based only on structural complexity by design of the user interface. In a more refined approach, all the parameters identified in [14] have to be considered as well.

Listing 4: Listing 3 with annotated complexity.

```
<ace name="simtd␣options␣screen" cpx="9.3">
 <root>
  <panel type="icon" cpx="0.8">
   <icon type="empty" cpx="0.1" />
   <icon type="empty" cpx="0.1" />
   <icon type="empty" cpx="0.1" />
   <icon type="empty" cpx="0.1" />
   <icon type="empty" cpx="0.1" />
   <icon type="empty" cpx="0.1" />
   <icon type="static" cpx="0.2" />
  </panel>
  <panel type="label" decoration="framed" cpx="3.4">
   <label text="true" icon="true" cpx="1.0" />
   <label text="true" icon="true" cpx="1.0" />
   <label text="true" icon="true" cpx="1.0" />
  </panel>
  <panel type="label" decoration="framed" cpx="2.4">
   <label text="true" icon="true" cpx="1.0" />
   <label text="true" icon="true" cpx="1.0" />
  </panel>
  <panel type="label" decoration="framed" cpx="1.7">
   <label text="true" icon="true" cpx="1.0" />
  </panel>
  <label text="true" icon="true" cpx="1.0" />
 </root>
</ace>
```

6. A CONTAINER LANGUAGE FOR ANNOTATED PRESENTATIONS

As shown in figure 1, we now have discussed the driver-related contribution to situation awareness and now will take a look at the presentation-related information, i.e., presentation complexity. By doing this, we start again at system situation awareness level 1 in our model.
Despite standardization efforts, there is still a large amount of available languages. Imposing a new, additional standard on top of existing approaches is not necessary.
As an alternative, I propose a wrapper-language as a container format for existing markups. Container languages are commonly defined as a meta file format whose specification describes how different data elements and meta data coexist in a computer file. The requirements for such a container language are simple:

1. It can contain different kinds of representation languages.

2. Meta-data on the presentation task necessary for the presentation toolkit can be stored in in.

3. Different alternative display strategies can be encoded (following the example of [2]).

4. Each display strategy can be annotated with a metric for preference as well as with a cognitive demand value, either as an overall value or in a more detailed way.

As one of the results of the presentation language survey, a clear tendency towards XML-based languages was identified. In accordance with that, the proposed meta-format introduced here–PTCL for PresTKContainer Language–is also XML based. A simple example is shown in listing 5.

Listing 5: A simple ptcl example wrapping two alternative display strategies of one presentation task.

```
<ptcl>
 <meta>
  <overallPriority value=70 metric="percent" />
 </meta>
 <displayStrategies>
  <strategy>
   <preference=1 />
   <demand=0.8 />
   <representation language="XY">
    [first variant of presentation task in language XY]
   </representation>
  </strategy>
  <strategy>
   <preference=2 />
   <demand=0.3 />
   <representation language="XY">
    [second variant of presentation task in language XY]
   </representation>
  </strategy>
 </displayStrategies>
</ptcl>
```

This example can be extended in various ways. Meta-data of the *presentation task* may also include *minimal presentation duration* or similar processing information specifying whether or not the scheduler may prepone or postpone this presentation on the temporal axis. The *displayStrategies*-Block may define some variables for recurring parts in more than one *display strategy* and the strategies itself can contain a more detailed demand definition as defined in section 4. Formalizing complexity values by annotating presentation corresponds to system situation awareness level 2.

7. SUMMARY

In this paper, we propose a number of elements for automotive presentation languages. First, we introduce the concept of System Situation Awareness (SSA), an extension of Endsley's influential model of (human) situation awareness. In order to obtain/implement SSA, we outlined two basic concepts: an ontological knowledge representation and management component (specifically KapCom) as well an approach on annotated complexity estimation (ACE). Finally, we propose a meta-language (container language) for annotated presentations that is considered to be a good starting point for discussing future standardization processes.

8. REFERENCES

[1] J. Antin, T. Dingus, M. Hulse, and W. Wierwille. The effects of spatial ability on automobile navigation. *Trends in Ergonomics/Human Factors*, 5(24):1–248, 1988.

[2] S. Castronovo. *Beyond the Push-Paradigm: Enhanced forms of Information Access and Novel Application Areas for Vehicle-2X Communication Networks.* PhD thesis, Computer Science Institute, Saarland University, Saarbrücken, Germany, 2013. to appear.

[3] Christoph Endres. *PresTK: Situation-Aware Presentation of Messages and Infotainment Content for Drivers.* PhD thesis, Saarland University, 2012 (to appear).

[4] G. Di Santo and E. Zimeo. Reversing guis to ximl descriptions for the adaptation to heterogeneous devices. In *Proceedings of the 2007 ACM symposium on Applied computing*, pages 1456–1460. ACM, 2007.

[5] R. Eckstein, M. Loy, and D. Wood. *Java swing.* O'Reilly & Associates, Inc., 1998.

[6] C. Endres. Real-time Assessment of Driver Cognitive Load as a prerequisite for the situation-aware Presentation Toolkit PresTK. In *Adjunct Proceedings of the 4th International Conference on Automotive User Interfaces and Interactive Vehicular Applications (AutomotiveUI 2012)*, Portsmouth, New Hampshire, USA, October 2012.

[7] C. Endres, D. Braun, and C. Müller. Prototyping a Semi-Automatic In-Car Texting Assistant. In *Proceedings of the 3rd Workshop on Multimodal Interfaces for Automotive Applications (MIAA 2011)*, pages 57–60, Palo Alto, CA, USA, February 2011.

[8] C. Endres, M. Feld, and C. Müller. A Layout-based Estimation of Presentation Complexity. In *Adjunct Proceedings of the 4th International Conference on Automotive User Interfaces and Interactive Vehicular Applications (AutomotiveUI 2012)*, Portsmouth, New Hampshire, USA, October 2012.

[9] M. Endsley. Toward a theory of situation awareness in dynamic systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 37(1):32–64, 1995.

[10] M. Endsley. Theoretical underpinnings of situation awareness: A critical review. *Situation awareness analysis and measurement*, pages 3–32, 2000.

[11] M. Feld and C. Endres. Sharing User and Context Models in Automotive HMI. In *Adjunct proceedings of the 2nd International Conference on Automotive User Interfaces and Interactive Vehicular Applications (AutomotiveUI 2010)*, page 10, Pittsburgh, PA, USA, November 2010.

[12] D. Heckmann, T. Schwartz, B. Brandherm, M. Schmitz, and M. von Wilamowitz-Moellendorff. GUMO–the general user model ontology. *User modeling 2005*, pages 428–432, 2005.

[13] D. Imbeau, W. W. Wierwille, and Y. Beauchamp. *Age, display design and driving perfomance*, chapter 16, pages 339–357. Taylor and Francis London, 1993.

[14] D. Imbeau, W. W. Wierwille, L. D. Wolf, and G. A. Chun. Effects of Instrument Panel Luminance and Chromaticity on Reading Performance and Preference in Simulated Driving. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 31(2):147–160, 1989.

[15] H. Krieger. A General Methodology for Equipping Ontologies With Time. In *Proceedings of the 7th international conference on Language Resources and Evaluation (LREC'10), ELRA*, 2010.

[16] R. Ovans and W. S. Havens. Intelligent mediation: an architecture for the real-time allocation of interface resources. In *Proceedings of the 1st international conference on Intelligent user interfaces*, IUI '93, pages 55–61, New York, NY, USA, 1993. ACM.

[17] R. E. Schlegel. *Driver mental workload*, chapter 17, pages 359–382. Taylor and Francis London, 1993.