Cognitive and Affective UI for Enhanced Usability

Sohail Rafiqi[a], Suku Nair[a] and Ephrem Fernandez[b]

[a]HACNET Labs
Department of Computer Science and Engineering
Southern Methodist University
Dallas, TX 75205, USA

[b]The Emotion and Sensation Research Group
Department of Psychology
University of Texas at San Antonio
San Antonio, TX 78249, USA

ABSTRACT

Dynamic working conditions along with the user's physical and emotional state have significant impact upon the usability and security of the system. As applications continue to provide increasingly sophisticated and complex functionality coupled with the widespread deployment of smart phones and tablets, critical information is at the fingertips of users. This level of accessibility substantially increases the possibilities of cyber security breaches. The impact of usability on reliability and security of the system are well understood, but what is not addressed is the impact of changing environment and cognitive load on the usability of the system. In this paper, we present the framework that constantly determines the user's affective state, cognitive load, and context information and makes real-time adaptations to the User Interface to enhance the usability and security of the system. The framework can operate in a multi or single-mode configuration. The multi-mode configuration considers all factors before making a decision, whereas the single-mode automatically learns to use only the available information.

Keywords: Cognitive and Context-Aware User Interfaces, Cognitive Load, Affective States, Usability, Reliability

INTRODUCTION

Computers are ingrained in our lives and in today's world of mobility and powerful tablets we have unprecedented access to information and decision-making (even in the most uncongenial environment). Though we constantly strive to ensure that the information shared during our interaction with the system remains effective, safe, enjoyable, and entertaining (Agah 2001), not all systems meet our expectations. The vast majorities of the systems still assume consistent working conditions and do not adapt to the user's situational influences and intrapersonal states (Hudlicka & Meneese 2002). Most of the user interfaces are designed for masses. Although many customized UI options are available, none of them provides dynamic customization based upon the user's affective states, cognitive load, and context.

A fluctuating work environment and the physical and intrapersonal state of the user can increase cognitive load on an individual's working memory (Morey & Morey 2011). This load refers to the burden or the impact that a task bears on human information processing (Sweller 1994). The mental resources of a user are directly proportional to the difficulty level of the task and presentation of the task. Since working memory is limited, the task imposes a load on the user during that interaction. This load is referred to as Cognitive Load. The working memory directly influences user performance. The higher load on working memory results in the lower performance (Barouillet, Bernandin & Portrat 2007).

The Dynamically Adaptable User Interface (DaUI) framework establishes the groundwork for dynamically adaptable applications capable of reacting to changes in the user's psychophysiological state as well as the working environment. The DAUI framework facilitates real-time security decisions based upon the combination of biometric data (e.g., pupillometric indices of cognitive load), environmental analysis, location, time and other details. The DAUI framework fundamentally changes the scope of cyber security to encompass whether the user has the mental capacity to carry on the task, and whether the user is to be allowed access based upon his/her current affective/cognitive state and/or usage pattern (Robins, Fraley & Krueger 2009). This framework is not intended to replace traditional access control (Ravi & Sandhu 1994) mechanisms but makes the argument that traditional access control is no longer sufficient. Effects of emotions and cognitive load on the user's ability to make decisions are widely known (Cao, Theune & Nijholt 2009)(Block & Hancock 2010). This framework enables applications to use psychophysiological sensing and context-awareness to selectively control access and provide user interface adaptations.

The DAUI framework determines cognitive load and affective states of the user using pupillometry (Laeng & Sirois 2012). The measurement of pupil diameter is referred to as Pupillometry. Specifically, changes in pupillary diameter correlate with cognitive load and affective change. External context is inferred from the location, time, background noise, and lighting. Factors such as time, location, and close proximity to others are used as input into security decisions. The DAUI framework includes the capabilities of providing Dynamic Ambient Intelligence (DaI) by constantly monitoring the user's environment. While doing so, it considers distractions (such as noise, crowd, and lighting) that have impact upon the user's security. For example, the application can hide sensitive information, or disable critical functionality if the user is fatigued or if non-authorized users are in close proximity. Additionally, the DAUI framework considers the user's location information. For example, applications can be restricted to only operate in certain locations, retract user access, or encrypt/destroy certain information if the device roams in an un-trusted area. Similarly, the time-of-day, and the day-of-the-week would contribute to security and application usability.

The framework constantly learns from non-intrusive surveillance and intelligence information and builds the knowledge-based system. The system shifts its operation to a heightened alert and vigilance state if it is outside the normal operating conditions. Higher vigilance results in lower tolerance of error. The objective is to customize the user interface to best meet the user's cognitive load and current working conditions to enhance security and usability. It is proven that the environment always plays a pivotal role in the performance of a user, but applications mostly ignore its impact. The cognitive load along with external factors, such as the environment, noise, crowd, and lighting contribute heavily in determining the usability and security of the system. These factors are taken into consideration as these elements impede the user's performance as well as pose potential security risk. The DAUI framework can disable critical functionality if the user's working environment is too distracting or unsafe. Location information is used to enhance both usability and security of the application. Sensitive applications can be restricted to operate in certain locations. If the device roams in an un-trusted area, information can be hidden, encrypted or completely destroyed. Similarly, time of day, and day of the week would contribute in security and application usability. The core of the DAUI Framework is the Dynamic Analysis Engine (DaE) that performs real-time data mining for the big data received from sensors.

COGNITIVE, CONTEXT-AWARE SECURITY FRAMEWORK

Overview

Current computing systems (desktop, laptop, and smart phones) are equipped with built-in devices such as cameras, microphone, and global positioning system (GPS) to operate with the framework. This system will use the built-in capabilities of the device to determine location, capture sounds, and perform other functions. The following is the list of high-level functions of the DAUI framework:

- Collect and analyze data from video, microphone, GPS.
- Determine the intensity, and duration of the collected data.
- Remove outliers and evaluate reliability.
- Represent behavior in a quantifiable manner.
- Perform continuous real-time gathering and analysis.
- Provide real-time feedback to the application.

Each application reacts differently to these events. For example, a GUI based application can increase the font size, modify the interface speed, or in some cases totally shutdown/disable certain parts of the interface for security reasons. A surveillance application can modify the monitoring interface to enhance visibility and focus.

Figure 1 provides a high-level view of the systems architecture as well as shows how applications fit into the DAUI Framework. Data are captured from a variety of sources and analyzed. Application developers use the newly proposed Markup Language (DaUIML) to develop their applications to enable dynamic adaptions based upon the events received from the Analysis Engine. The Analysis Engine generates events after real-time analysis of the data captured using built-in sensors.

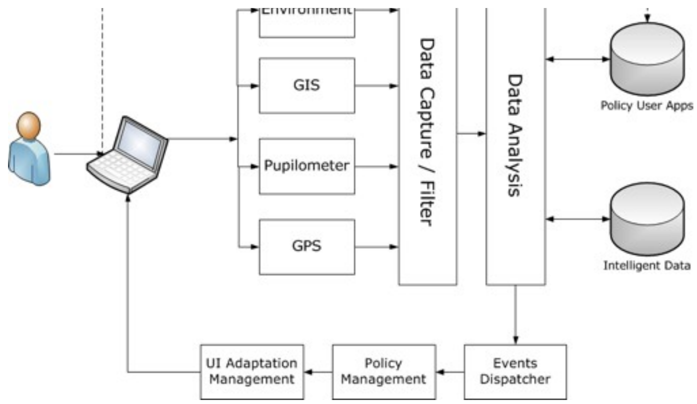

Figure 1: The DAUI Process Flow

DAUI Framework Contributions

The DAUI Framework has the following contributions

1. Dynamic Analysis Engine that captures, analyzes, and filters all data at real-time using built-in devices.
2. Real-time determination of cognitive load and affective states using pupillometric measurements.
3. Dynamic user context determination.
4. A dynamic User Interface Model (MIDM).
5. Enhanced XML-compliant (DaUIML) language that enables dynamic adaptations based upon biometric, ambient, time and location data.

DAUI Architecture

The fundamental concept of the DAUI framework is built around speed and the ability to easily extend functionality. The design of the DAUI framework is intended to be modular, efficient, and interoperable, and provides a generic infrastructure for the dynamically adaptable user interface for any application. All inputs would be aggregated from multiple devices such as: traditional/virtual keyboard, mouse, touch panel, front/rear camera, microphone input.

The framework provides an additional layer between the application and underlying operating system, and facilitates a seamless integration with the existing applications. The input parameters and subsequent desired actions would be modeled and represented as state machines and push down automata. Moreover, the modularity of the design facilitates using the framework in variety of deployment scenarios. Figure 2 illustrates the high-level proposed architecture of the DAUI. There are three basic components of the DAUI framework:

i) Analysis Engine

ii) Cognitive Load and Context determination,

iii) XML-compliant grammar, and

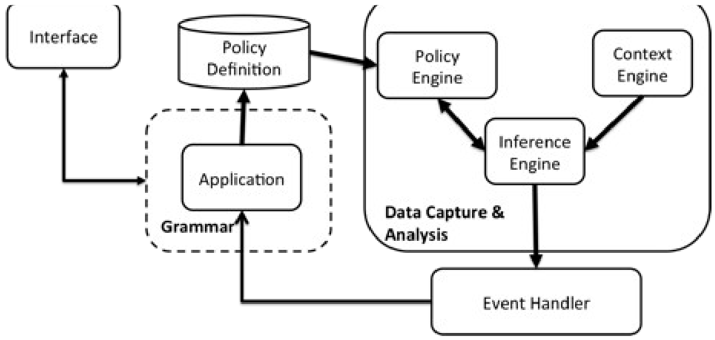

Figure 2: DAUI System Overview

The Dynamic Analysis Engine (DaE) is an integral part of DAUI framework, and it extends the framework's capabilities to harness the potential of external devices and various computing resources, and ensure that they have a real-time adaptive user-interface. Additionally, DaE can be deployed as a distributed system utilizing numerous computing resources available via a cloud based computing engine. Moreover, multiple Analysis Engines can run simultaneously in a single deployment installation mining data from various sources to ascertain features. The DaE transformational and adaptation layer facilitates transmission and reformatting of relevant key data points from one system to another. This is required to ensure harness of the large amount data available from application past usage history, interactions of multiple users of the application and behavioral predictions of the usage pattern.

COGNITIVE LOAD

Any complex application requires extensive resources from users to build and maintain situation-awareness. This is certainly true in any complex system. Endsley & Bolte (Endsley & Bolte, 2003) describe what they call "situation-awareness demons"; these are the factors that negatively impact upon situation-awareness in many systems. This includes memory trap, workload, anxiety, fatigue, boredom, and other psychological states. The memory trap is directly related to the limited working memory wherein chunks of data are stored temporarily prior to information processing. Since situation-awareness requires greater use of the working memory, any system that overloads working memory directly obstructs user performance.

The other impediment of situation-awareness is the individual operating environment (Endsley & Bolte, 2003), in which some of the common conditions are fatigue and anxiety. All of these factors directly impact the ability of the individual to process information. The system has to be designed in such a way that all these limitation can be countered. We make the argument that the mobility and ubiquity of computing devices make it very challenging to overcome these impediments in a statically designed system.

Previously, studies have been conducted to determine ways to reduce cognitive load in the multi-media learning environment (Mayer & Moreno 2003)(Riddle 2002). In addition to providing great insight into ways to enhance usability, analysis also asserts that cognitive load is the essential component of human machine interaction and learning (Zhang & Rau 2009). Riddle (Riddle 2002) proposed a framework to reduce the cognitive load associated with navigational activities in e-commerce domain. E-commerce applications are extremely competitive and need to keep the user engaged in the buying experience if users experience higher cognitive load that can have a direct impact on their buying experience and result in loss revenue.

There are numerous ways that a human computer interface can cause cognitive load to increase (Paas, Renkl & Sweller 2003). This includes both essential processing load and incidental-processing load. Incidental workload refers to the processing that the brain has to perform, and is not related to the core function.

Measuring Cognitive Load

Measuring accurate cognitive load has always been a challenging part of the Cognitive Load Theory (CLT) (DeLeeuw 2008), but this is also the most important aspect for providing real-time cognitive aware security modulation. Incorrect or untimely cognitive load determination can have adverse effects of adding more cognitive load rather than reducing it. According to the Cognitive Load Theory there are three types of cognitive loads (Sweller 1994) Intrinsic, Extraneous, and Germane.

- Intrinsic – The Intrinsic load is related to the difficulty of the task at hand, and lack of familiarity with the task. This load is the thinking part of the CLT and mainly considered as unalterable. This is because the fundamental complexity of the task cannot be modified. Though the difficulty of the task cannot be modified, it can be broken down to reduce the load. Complexity of any task is directly proportional to the number of different elements considered to complete the task.
- Extraneous – The Extraneous load is the material part of the cognitive load. It is directly related to the presented information. **Note**: Information presented in a poor manner adds to the load.
- Germane -- The Germane load signifies the effort required to process the information and construct schema and automation (Sweller 1994)

This DAUI framework primarily focuses on reducing the Extraneous load and increasing the Germane load by framing and modulating information that is more intuitively comprehensible. The automation of schema generation will help in minimizing the load on the working memory. Our objective in this study is to measure the user’s current cognitive load, make dynamic adaptations to the presented information, and thereby reduce the working load. Methods of measuring Cognitive load fall into two categories, Subjective (Self-report)(Robinson & Clore 2002) and Objective measures (DeLeeuw 2008)(Chen, Epps & Chen 2011)(Klingner 2010).

TABLE 1: COGNITIVE LOAD MEASUREMENT CATEGORIES

Measurement	**Category**
Self-Report	**Subjective**
Physiological behavioral, Task performance	**Objective-indirect**
Brain Activity, Dual-Task performance	**Objective-direct**

These methods of measuring Cognitive Load are useful and provide different levels of performance and accuracy as defined in (Chen, Epps & Chen 2011). Appropriate selection of the method however depends upon the requirements. Self-report is a widely acceptable and extremely common test to determine the emotional experiences of a user but it relies upon the user providing accurate and timely information. In certain mission applications, self-report can be used to determine the cognitive load but it is not feasible where continuous real-time cognitive load determination is required.

Another area of research pertains to the use of psychophysiological response-based measurements in determining the cognitive load on the user (Iqbal, Zheng & Bailey 2004) (Klingner 2010) (Haapalainen, Forlizzi & Dey 2010). Various studies have been conducted to support the conclusion that variations in the mental load generate involuntary psychophysiological responses (MacLachlan & Howland 2002)(Rubin 1980)(Richman, McAndrew, Decker & Mullancy 2004).. These psychophysiological responses can range from heart rate variability, galvanic skin response, and pupillary dilation (Haapalainen, Forlizzi & Dey 2010). In the DAUI framework we use pupillary responses (pupil dilation) as a means to measure cognitive load. The reason we chose pupil dilation as the measure of cognitive load is that it provides real-time information and does not require any explicit actions from the user. Moreover, depending on the methods of measuring the pupil dilation it can be very non-intrusive (Iqbal, Zheng & Bailey 2004) (Klingner 2010). Much research has been done to establish the correlation between cognitive load and pupil dilation. Iqbal et al (Iqbal, Zheng & Bailey 2004) conducted a user study to successfully determine the use of pupillometry for measuring real-time mental load while the user is interacting with the computer. Similarly Klingner

(Klingner 2010) conducted an extensive study to validate the use of a remote eye-tracking device to accurately measuring real-time cognitive load.

The DAUI framework measures pupil dilation based upon the average change in pupil dilation (ACPD). The user's baseline pupillary diameter is established against which any changes are measured. Unlike strictly task-evoked pupillary response measurement (Klingner 2010) where the baseline is taken at the beginning of the task, the DAUI framework learns and stores the baseline pupillary diameter and uses this to measure variation in pupil size. For example, if the user starts to use the system when s/he is fatigued, the DAUI framework will capture the user's current pupillary diameter and compare it to the stored baseline pupillary diameter to conclude that the user is fatigued. This is important as this can result in the DAUI framework precluding the user from using the system altogether or blocking that part of section of the system.

MacLachlan & Howland (MacLachlan & Howland 2002) conducted a study to determine the normal pupillary diameter for subjects ranging from birth to 19 years. As a result of this study approximate formula based upon subject age were developed. The DAUI framework uses these formulae to validate the captured baseline pupil dilation.

- The approximate male pupil diameter:
  Pupil Diameter (mm) =
  $5.83 + 0.181 * \text{age in years} – 00.53 * \text{age in years}^2$

- The approximate female pupil diameter:
  Pupil Diameter (mm) =
  $5.40 + 0.285 * \text{age in years} – 00.53 * \text{age in years}^2$

Correlation between Pupil Size Variance and Cognitive load/Affective States

Correlating the variance in pupil size with (i) cognitive load, (ii) emotional and (iii) physical state is an integral part of the DAUI framework. The Average Change in Pupil Dilation determined dynamically is compared against pupil variance guidelines. The following pupil diameter changes described in Klinger (Klingner 2010) are used as a benchmark to determine the cognitive load:

TABLE 2: PUPIL DILATION TO COGNITIVE LOAD CORRELATION

Average Pupil Dilation	Cognitive Load
0.17 mm	Light
0.21 mm	Medium
0.27 mm	Heavy

Furthermore, the average pupil size variance can also be related to other user's conditions such as fatigue, alcohol level, etc. (Tryon 1975)(Rubin 1980)(Richman, McAndrew, Decker & Mullancy 2004).

DYNAMIC GRAMMAR

The DAUI framework defines a new XML-based grammar, Dynamically Adaptable User Interface Markup Language (DaUIML). This is used to develop cognitive and context-aware applications that can dynamically alter behavior based upon external events. The DaUIML leverages some key characteristics of UIML and describes the user-interface in abstract declarative terms. Using UIML (Oasis UIML Specification)(Phanouriou 2000), a one-to-one tag mapping for various existing graphical frameworks could be formulated to structurally represent the user-interface for an application and define static components and associated properties. This includes the generic event processing mechanism utilized in the current frameworks from various input devices. Furthermore, this may include real-time Web/Mobile application characteristics that involve updating the user-interface based on external events such as updating the backend Data store.

The DaUIML extension to UIML provides a true dynamic nature to an application using the events generated by the dynamic analysis engine (DaE). This would mean that an application developer could use the DaUIML as a meta-language to specify policies to update any of the properties associated with existing user interface components, in terms of structure, style, content and behavior using events generated by the DaE. This provides application developers a finer grain control over the individual UI components. The following table shows a small subset of DaUIML tags:

TABLE 3: DAUIML TAGS

DaUIML Tag	Purpose
<application-context>	Represents user/system state throughout application life cycle.
<dae-event>	Represent event from the analysis engine in meta-attribute format.
<dae-core-policy>	Specify policy/action as the application receives an associated event.

MESSAGING MODEL

The messaging model deals with the unidirectional relay of events from sensors to the analysis engine as well as bidirectional exchange between the analysis engine and the application. In order to facilitate high throughput communication messages are exchanged in a fixed format with labeled number of fields. The messages are divided into mandatory and optional fields such as compression, encryption and versioning. In order to facilitate faster index based access all enumerated fields in the message can be encapsulated as a single structure. Additionally, message specification can include attribute/field validation. Examples for a basic binding for a user-defined complex data-type is as follows:

TABLE 4: BASIC BINDING

```
<structure field-ordering-binding=’strict’ field-count=’x’>
  <field type=’signed-integer-16’ field-name=’device-detected’
verification=’optional’ defatul=’1’ range-check=’true’ min-value=’1024’></field>
 <field type='string'  field-name='device-name' verification =  'not-null' > </field>
  <field type='uuid'  field-name='device-identifier' verification ='not-null' ></field>
</structure>
```

Using the above technique the application developer can introduce custom data-types that can represent events, and exchange request/response between DaE and embedded/external devices.

Policy Management

The policy management provides a dynamic capability to a user interface application by introducing meta-tags related to the application state and complex event processing. Policies are the intrinsic part of user-experience, and enhance and update the user-interface components to assist a user in operating the application with utmost efficiency and clarity. In addition, the updates in each user-interface component are guided by the defined application policy. The policy enforcement is aided by the real-time inputs from the analysis engine, which ensures that user-interface representation constantly meets the expectation of the application designer with respect to the usage criteria. This is required to guarantee that the user-interface updates and restructuring are always within the specific implementation boundaries.

Sample Policy

Following is the abbreviated example of a very simple policy that specifies policy to make patient prescription protected (un-editable):

TABLE 5: SAMPLE POLICY

```
<event>
  <dae-core-event class=”user-cognitive-load”>
  <dae-event-source-name=”dynamic-analysis-engine” level=”1”
</dae-core-event>
</event>
 <op name=”greater-than”>
   <property event-class=”user-cognitive-load” name=”current load”
     <! – Cognitive load measured as Low=1, Med=2, High=3 >
       <constant value=’1’ />
  </op>
</op>
</condition>
  <action>
    <! Make sensitive information uneditable >
    <property part-name="PatientPrescription" name="editable">false</property>
   <! Update property “Application-State” to value returned from analysis engine -->
     <property part-name="Application-State" name="user-profile-state">
    <call name="ApplicationFunction.updateApplicationState">
                                    <param><property       part-class="application-state"
    name="current-state"/></param>
            </call>
          </property>
     </action>
```

EXTERNAL EVENT INTERFACE

Dynamic Analysis Engine (DAE) can issue notifications to an application requiring real-time analysis services from various sources. The information to be included in these notifications and event can be configured as per application requirement. Also, apart from these the user-interface application developer can optionally include additional information as per application criteria for each of these notifications. These application notification feedback responses for each respective event is intended to be stored in the user/application knowledge base and used later for further enhancing the overall user experience. These events are used to update the following aspects of the user-interface application:

- Application content
- Component Layout/presentation
- Application Navigation
- Application logic/control flow (additional security criteria)

The Dynamic Analysis Engine [DAE] is intended to provide in depth analysis of events processed from multiple sources in real-time. An application using the services of Dynamic Analysis engine can use more than one DAE engine to provide analysis from multiple and/or remote sources as well. Additionally, an application like DaUIML rendering application could process events involving real-time user/group administration, enterprise application usage policy updates and various diagnostic checks. These events could also be generated from other administrative application apart from DAE. The application using Dynamic Analysis Engine service can establish a real-time event feedback loop by providing update application state and updates which could be required in application content, navigation and behavioral aspect as per the usage criteria as part of response sent after processing the DAE events.

The Dynamic Analysis Engine [DAE] utilizes JSON (JavaScript Object Notation) specification for interaction with external application. The JSON (Crockford 2006) specification is a lightweight data-interchange format. We can use JSON to represent structured information in a human readable form. Additionally the information could contain additional tags to validation message contents using checksum and enforcing strict field ordering. Along with message checksums the message contents could be encrypted as well. Following is an excerpt of the JSON schema definition for DAE event request/response:

TABLE 6: JSON SCHEMA


```
{
    "event-name": "Generic event name for informational purpose only",
    "id": {
        "type": “UUID Universally unique identifier",
        "description": "Distinguishes each individual event instance",
        "required": true
    },
    "source-identifier": {
        "type": “UUID Universally unique identifier",
        "description": "Distinguishes event generator module",
        "required": true
    },
    "destination-identifier": {
        "type": “UUID Universally unique identifier",
        "description": "Distinguishes event target module, can be a broadcast message",
        "required": true
    },
    "event-type": {
        "type": “Enumerator",
        "description": "Event type either of request/response or notification",
        "required": true
    },
```


APPLICATIONS

There are numerous applications where DAUI framework can be used to build applications intending to provide dynamic adaptation to enhance usability and security.

Homeland Security Applications

The DAUI framework can turn normal video surveillance into an advanced eye performing real-time reconnaissance to identify suspicious behavior, unaccompanied baggage, cross border activities, behavioral patterns, and identify subjects, etc. This system could automatically sift through big data and provide timely alerts. The adaptations to the user interface of the monitoring system are made to reflect the cognitive load of the monitoring agent. The system can be deployed in a centralized as well as distributed configuration.

Medical Instrumentation

All of the current medical system are all based upon the static interfaces and are prone to costly errors and inefficiencies. In addition, medical systems such electronic medical records (EMR) are vulnerable to data integrity and confidentiality issues. A Physician’s job of entering patient’s records, reviewing test results, prescribing medicine or simply finishing up the paper work at the end of the long working day, can be a frustrating and time-consuming experience (Pertet & Narasimhan 2005). This can lead to costly human errors. Physicians told Health and Human Services that the “usability does not appear to be the focus of these EMRs” (Fiegl 2011). Amongst other things these user errors can also compromise data integrity. The DAUI framework can enhance the security of the EMR system by granting selective access depending upon whether the user is tired, angry, asleep, etc. In addition, the DAUI framework can be used to make user interfaces of other medical instruments for dynamic such that physicians and nurses can be prevented from making mistakes.

EXAMPLE APPLICATION

Since the development of the complete DAUI framework involves many components (such as Grammar, Analysis Engine, Input and Analysis of real-time events information, context determination, and real-time UI adaptation) we have divided the implementation into multiple phases to facilitate the development. The first phase is focusing on a simple application using the CCAML grammar that receives events such as multiple users detected, cognitive load high, etc. from a test driver simulating the Dynamic Analysis Engine. The test driver uses the JSON interface described above to communicate information from the test driver to the application. The current application, which is simply a registration form, has rules defined to adapt the UI based upon these events. A basic registration form used for entering personal details for an applicant enrolling for a service. The application comprises of a set of paired label and text-field components displaying individual's name, date of birth, social security number, address and additional personal notes. The example application contains the following policy criteria:

- Use-case I
  - If user's cognitive load is high then system makes critical fields un-editable.
- Use-case II
  - If the system detects other people in proximity who can possibly view information on user's screen, system automatically hides critical information.
- Use-case III
  - The framework also adjusts to the outside lighting by changing the background colors.

Figure 3 shows one such case where the user has full access to all information when application is first started. As soon as application receives the cognitive load event, the application makes critical information un-editable as shown in figure 4. Any of the policies specified here are strictly arbitrary, as it depends upon the how application developer would like the application to respond to these events. The policies are customized to an application to best fit the business needs. A similar event in a different application may result in a totally different behavior.

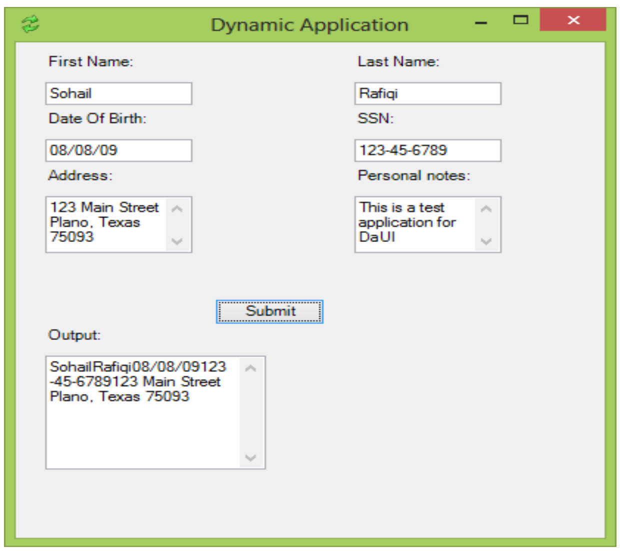

Figure 3: Registration Window

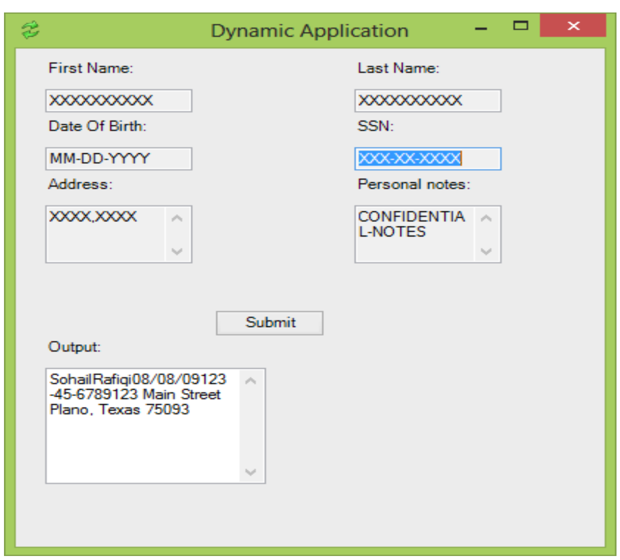

Figure 4: Critical information un-editable

The following is the excerpt of the code showing three use cases.

TABLE 7: APPLICATION CODE

```
<?xml version=”1.0”?>
<!DOCTYPE ccaml PUBLIC "-//DX-CORE//DTD CCAML 1.0a Draft//EN" http://dxcore.site/dtds/CCAML1_0a.dtd>

    <!-- dynamic behavior introduced using dx-core-events -->

     <!-- USE-CASE I :: If the user's cognitive load is high with the user location being out of office and time is outside
normal working hour then restrict the ability for user to edit the sensitive information -->
    <behavior>
      <rule>
        <condition>
          <op name = "and">
            <event>
              <dx-core-event class="user-cognitive-load">
                <dx-event-source name="dynamic-analysis-engine" level="1"/>
              </dx-core-event>
            </event>
            <op name ="greater-than">
              <property event-class=”user-cognitive-load” name=”current-load”/>
              <!-- Cognitive load measured in values between 1-10 with 10 being the highest -->
              <constant value="7.5"/>
            </op>
          </op>
          <op name = "and">
            <event>
              <dx-core-event class="user-location">
                <dx-event-source name="dynamic-analysis-engine" level="1"/>
              </dx-core-event>
            </event>
            <op name ="not-equal">
              <property event-class=”user-location” name=”current-location”/>
              <constant value="registered-office-work-area"/>
            </op>
          </op>
          <op name = "and">
            <!-- current device system/local time is either less than 08:00 AM or greater than 07:00 PM -->
            <op name ="less-than">
                <property class=”device-properties” name=”system-time” qualifier-format="24 Hours format"access-
format="HHMM"/>
              <constant value="0800"/>
            </op>
            <op name = "and">
              <op name ="greater-than">
                <property class=”device-properties” name=”system-time” qualifier-format="24 Hours format"access-
format="HHMM"/>
                <constant value="1900"/>
              </op>
            </op>
          </op>
        </condition>
        <action>
          <!-- Make sensitive information uneditable -->
          <property part-name="SocialSecurityNumberInput" name="editable">false</property>
          <property part-name="PersonalNotesInput" name="editable">false</property>
          <!-- update property “Application-State” to value returned by Dx-Core method -->
          <property part-name="Application-State" name="user-profile-state">
            <call name="ApplicationFunction.updateApplicationState">
              <param><property part-class="application-state" name="current-state"/></param>
             </call>
          </property>
        </action>
      </rule>
    </behavior>
```

CONCLUSION

In this paper we presented a multi-modal, dynamically adaptable framework that dynamically adjusts a user interface based upon the user’s cognitive load, affective states and the context. The high-level architecture showed the use of pupillometry to determine the cognitive load. We provided information about the new grammar, external event interface as well as the example application currently under development along with the code. In addition, we also discussed two potential applications for the framework. We argued that the ubiquity of technology adoption and our total reliance on technology has fundamentally changed the landscape of security as well as our interactions with computers. Moreover, the usability of the system directly impacts upon the reliability and security. The DAUI framework constantly adjusts the user interface to ensure consistent usability. These cyber attacks come from both external hackers as well as internal trusted sources. They can be both malicious in nature as well as accidental, but either way the costs are extremely high. We do not have enough resources to rely on humans to do all security screening and behavior detection. We need a system that can covertly perform that function to identify potential threats. The presented framework is a perfect union of two disparate sciences, computer science and psychology. The combination of understanding human mental and emotional characteristics and real-time data mining of large amount of data makes this framework unique.

REFERENCES

Agah, A., 2001. Human interactions with intelligent systems: research taxonomy. Computers & Electrical Engineering, 27, 71–107.

Bin Zhang, Pei-Luen Patrick Rau, G. S. (2009), 'Design and evaluation of smart home user interface: effects of age, tasks and intelligence level', Behavior & Information Technology 28(3), 239-249.

Bruno Laeng, Sylvain Sirois, G. G. (2012), 'Pupillometry: A Window to the Preconscious?', Perspectives on Psychological Sciences 7(1), 18-27

Cao, Y.; Theune, M. & Nijholt, A. (2009), “Modality effects on cognitive load and performance in high-load information presentation, in 'Proceedings of the 14th international conference on Intelligent user interfaces', pp. 335--344.

Charles Fiegl, Complexity of EMRs discourages new users, doctors tell HHS panel, Amednews.com April 29, 2011

Chen, S.; Epps, J. & Chen, F. (2011), “A comparison of four methods for cognitive load measurement, in Cognitive Load, ed., 'Proceedings of the 23rd Australian Computer-Human Interaction Conference', pp. 76--79.

Crockford, D. ‘Request for Comments 4627 -- The application/JSON Media Type for JavaScript Object Notation (JSON)’ Network Working Group 2006

DeLeeuw, K.E., &. M. R. (Journal of Educational Psychology 100 (1): 223P234), '"A comparison of three measures of cognitive load: Evidence for separable measures of intrinsic, extraneous, and germane load."

Endsley M, Bolte B, J. D. (2003), 'Designing for situation awareness: an approach to human-centered design. Taylor & Francis, London.

Haapalainen, E., Kim, S., Forlizzi, J. F., and Dey, A. K. Psycho-physiological measures for assessing cognitive load. In Proc. UbiComp 2010, 301-310.

Hudlicka, E. & Mcneese, M. D. (2002), 'Assessment of user affective and belief states for interface adaptation: Application to an Air Force pilot task', User Modeling and User-Adapted Interaction 12(1), 1--47.

Iqbal, S. T.; Zheng, X. S. & Bailey, B. P. (2004), Task-evoked pupillary response to mental workload in human-computer interaction, in 'CHI'04 extended abstracts on Human factors in computing systems', pp. 1477--1480.

Klingner, J. M. (2010), 'Measuring Cognitive Load During Visual Tasks by Combining Pupillometry and Eye Tracking', PhD thesis, Stanford University.

MacLachlan C, Howland HC. Normal values and standard deviations for pupil diameter and interpupillary distance in subjects aged 1 month to 19 years. Ophthalmic Physiol Opt 2002; 22:175-82.

Mayer R, Moreno, R, 'Nine Ways to Reduce Cognitive Load in Multimedia Learning', Educational Psychologist 2003, 38(1), 43-52.

OASIS UIML Specification https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=uiml

Paas F., A. Renkl, J. Sweller, “Cognitive Load Theory and Instructional design: Recent Developments”, Educational psychologist, 2003 - Taylor & Francis

Pertet, S. Narasimhan, P. Causes of Failures in Web Applications, Parallel Data Laboratory, Carnegie Mellon University, CMU-PDL-05-109

Phanouriou, C. (2000), 'UIML: A Device-Independent User Interface Markup', PhD thesis, Virginia Polytechnic Institute and State University.

Pierre Barrouillet, Sophie Bernardin and Sophie Portrat, 'Time and Cognitive Load in Working Memory', Journal of Experimental Psychology:, Learning, Memory, and Cognition, 2007, Vol. 33, No. 3, 570–585.

Ravi S. Sandhu, P. S. (September 1994), 'Access Control: Princples and Practice', IEEE Communications Maga, 40-48.

Richard A. Block, Peter A. Hancock, D. Z. (2010), 'How cognitive load affects duration judgments: A meta-analytic review', Acta Psychologica.

Richard Morey, Candice Morey, 'WoMMBAT: A user interface for hierarchical Bayesian estimation of working memory capacity', Behav Res (2011) 43:1044-1065

Richman, J. E.; Golden McAndrew, K.; Decker, D. & Mullaney, S. C. (2004), 'An evaluation of pupil size standards used by police officers for detecting drug impairment', Optometry-Journal of the American Optometric Association 75(3), 175--182.

Riddle, R. (), 'Reducing the user's cognitive burden using an improved navigational interface', Dissertation – Colorado Technical University.

Robins R.W, Fraley R.C, Krueger R.F., Handbook of research methods in personality psychology (pp.224-239) New York: Guilford Chapter 13.

Robinson, M. D. & Clore, G. L. (2002), 'Belief and Feeling: Evidence for an Accessibility Model of Emotional Self-Report', Psychological bulletin 128(6), 934.

Rubin, L. S. (1980), 'Effects of Alcohol on Autonomic Reactivity in Alcoholics – Pupillometric Studes', Journal of Studies on Alcohol 41(7), 611-622.

Sweller J., “Cognitive laod theory, learning difficulty, and instructional design” Learning and Instructuction, 1994 4(4), pp 295-312.

Tryon, W. W. (1975), 'Pupillometry: A survey of Sources of Variation', Psychophysiology Methodology 12(1), 90-93.