Contextual Car Driver Interface

ECE Seminar

Daniel Siewiorek & Asim Smailagic

20 September 2001

Outline

- Motivation and Overview
- Goals
- Scenario
- Car-Driver Interface
- Software Architecture
- Agents
- System Architecture
- Cognitive Models
- Conclusions

Motivation and Overview

- User studies and driver observations to ascertain driver behavior patterns (40 types of activities while driving were identified)
- Visionary scenario captured:
  - Features desired by drivers, as documented by interviews with drivers
  - Injection of new technologies to enhance driving
- Multimodal user interface with information hierarchy
- Implemented publish/subscribe software architecture
- Context aware platform
- Intelligent agents
- Capitalizing on extensive experience in system design and integration, multi-modal interfaces, and human-computer interaction

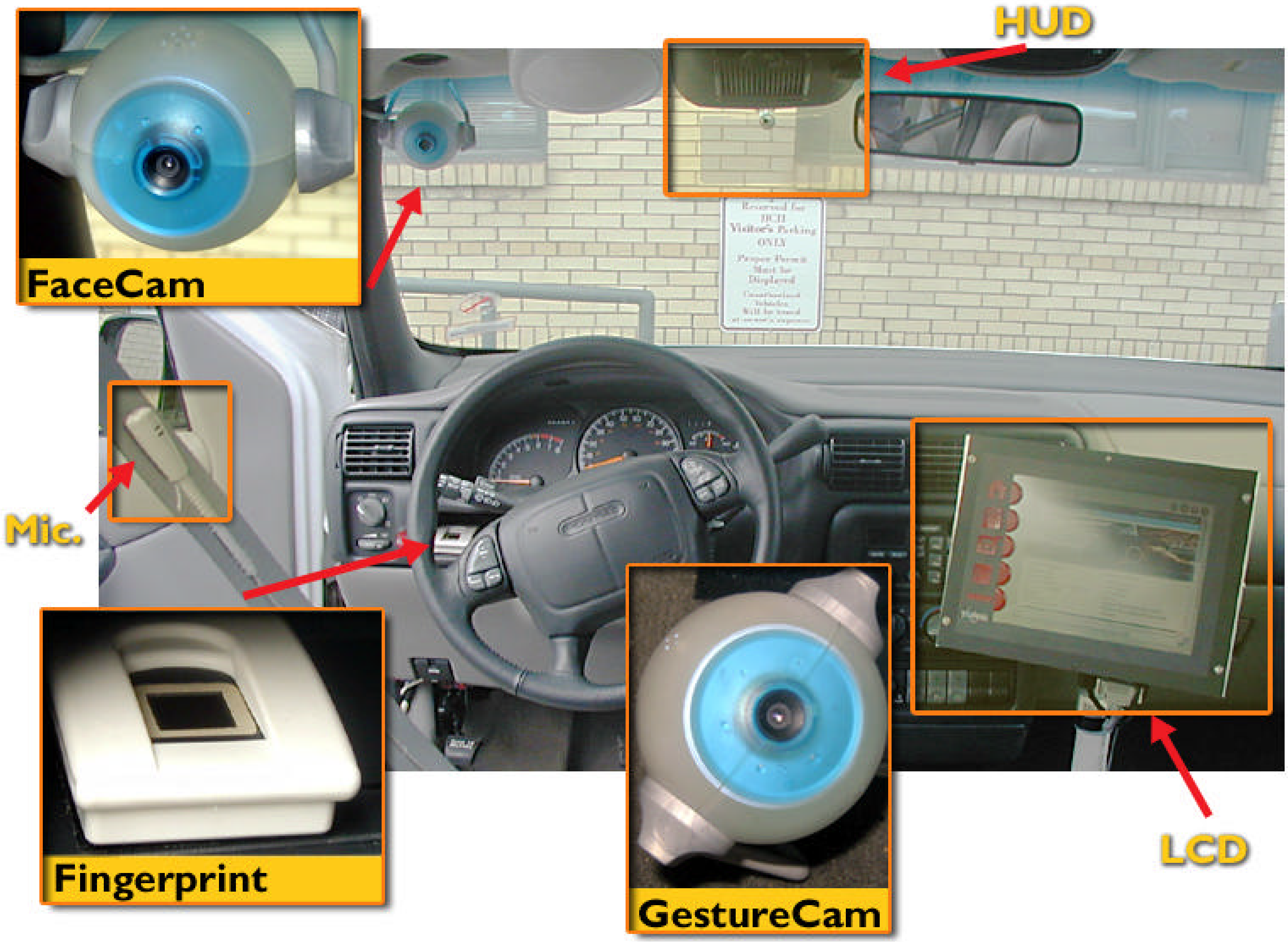
Goals

- Make driving a safer and more useful experience.
- Develop a context aware environment which links information from many sources: the driver’s life, current location in time and space, surroundings, entertainment media, the Internet, etc.
- Combine a smart car environment and monitoring of driver state, with a wide range of input-output modalities.
- Develop intelligent agents offering capabilities such as: learning about driver's behavioral patterns, proactively providing help when needed.

Introduction | Design/HCI | Software | Electronics

Driver Information GM Interface Project

Rapid Prototyping of Computer Systems Spring 2001 Carnegie Mellon University



GM/CMU Companion driver interface system

A Scenario of Use

Tom

Age: 32

Recently moved to Pittsburgh in order to pursue his career with a marketing firm

Busy schedule causes Thomas to spend large amounts of time driving in his car

Uses PDAs, a cell phone and a PC

Tom enters the car and is authenticated using fingerprint

The car greets Tom and downloads his address book, to do list, and schedule from his PDA.

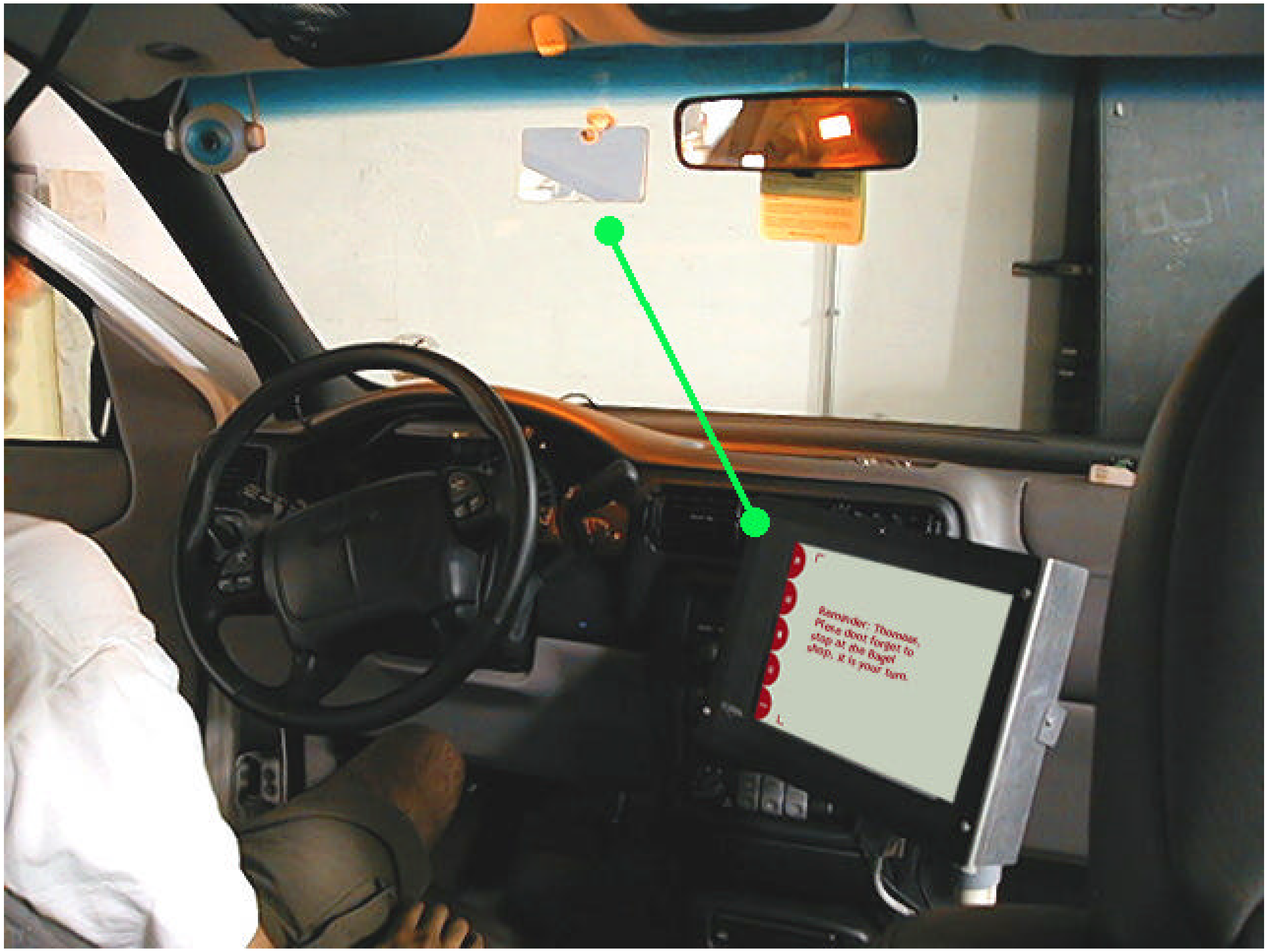
Introduction | Design/HCI | Software | Electronics

Tom's schedule is displayed on the LCD. He reviews it, and begins to drive to work.

As he drives to work, he receives an urgent notification on the heads up display

His fuel is running low. Tom stops at a gas station.

The Space/Time Agent, using GPS data, notices that Tom is near the bagel store.



The speech synthesis system tells Tom that it is his turn to buy bagels, and a message is displayed on the LCD panel.

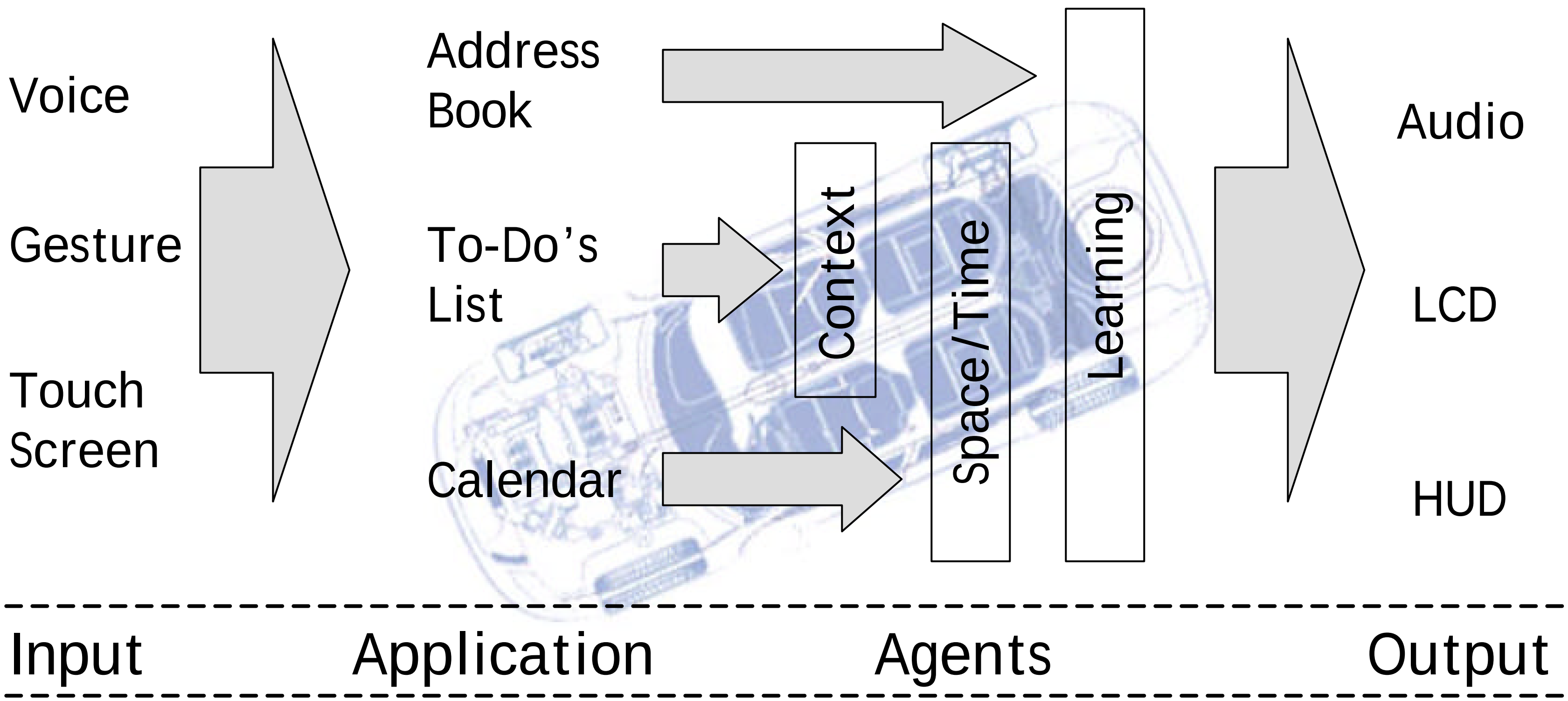
Using the integrated CDPD modem and web browser, Tom downloads the bagel store's menu, then uses the USI speech dialogue system to place a call.

Thomas picks up the bagels from the bagel store.

Input/Output Information Path

- Driver has a task on the To Do list to pick up bagels.
  - The driver presses on the touch screen to check off the completed task
  - To Do program removes the task from its database
  - Space/Time Agent removes it from the list of tasks it is monitoring
  - The task is removed from the display

Input/Output Information Path

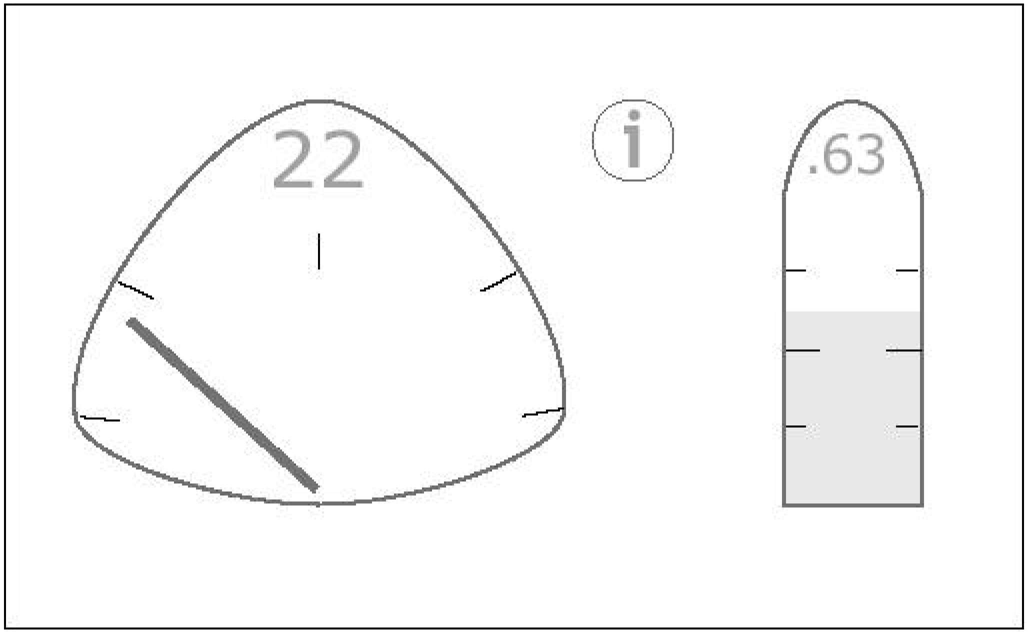


- Gesture Recognition: Pause/Stop/Start the system
- Face/Fingerprint Recognition: Driver Identification

Information Hierarchy

- **LCD**
  - Low priority information
  - Stationary: extensive user interaction
  - In motion: minimal user interaction
- **Voice**
  - Medium priority information
  - Limited user interaction
- **HUD**
  - High priority information
  - No user interaction

HUD Interface



Speedometer, Driver Alert and Gas Level Indicators

LCD Main Screen

Home

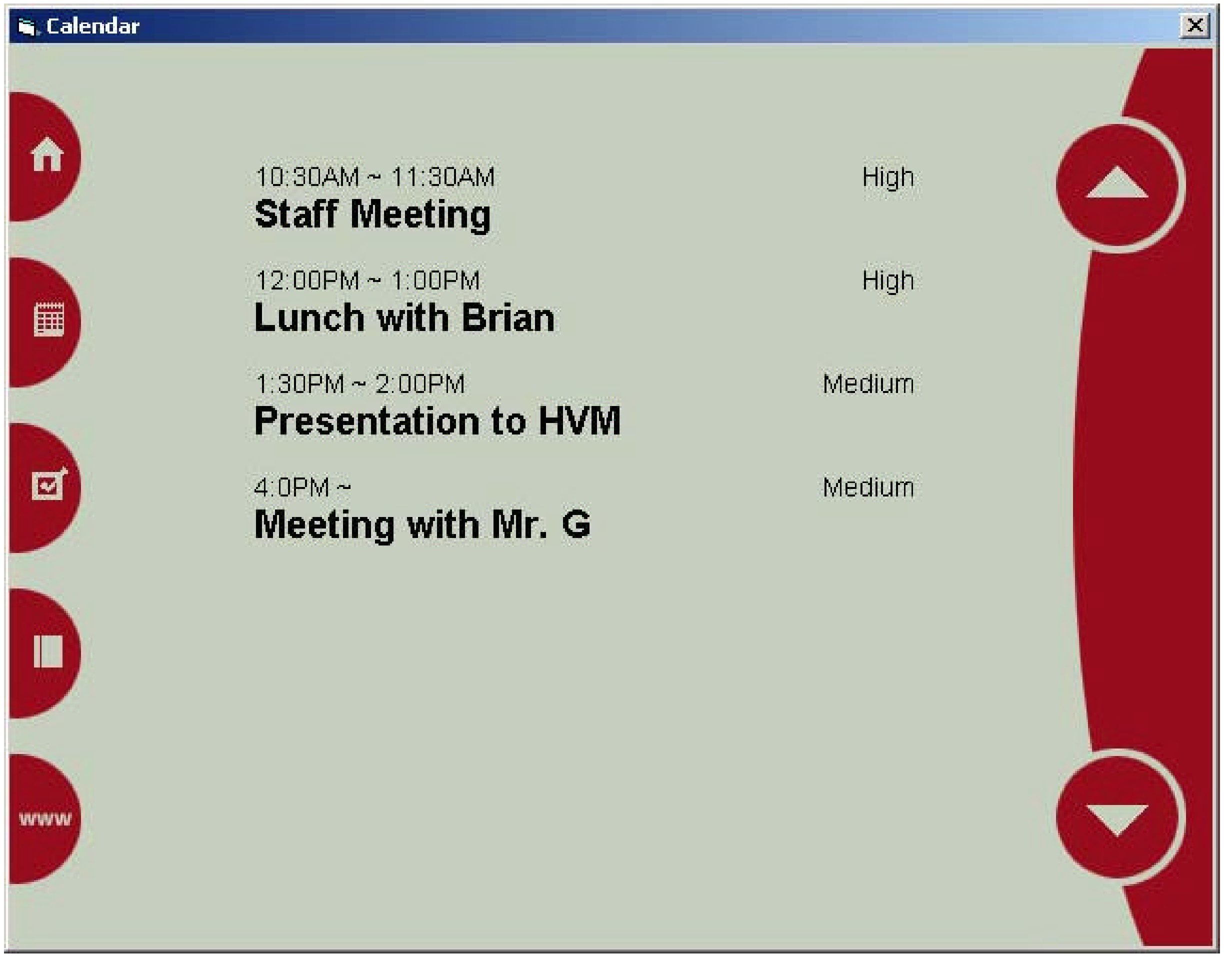
Calendar

To Do List

Contacts

Web

LCD Calendar Screen



USI – Universal Speech Interface

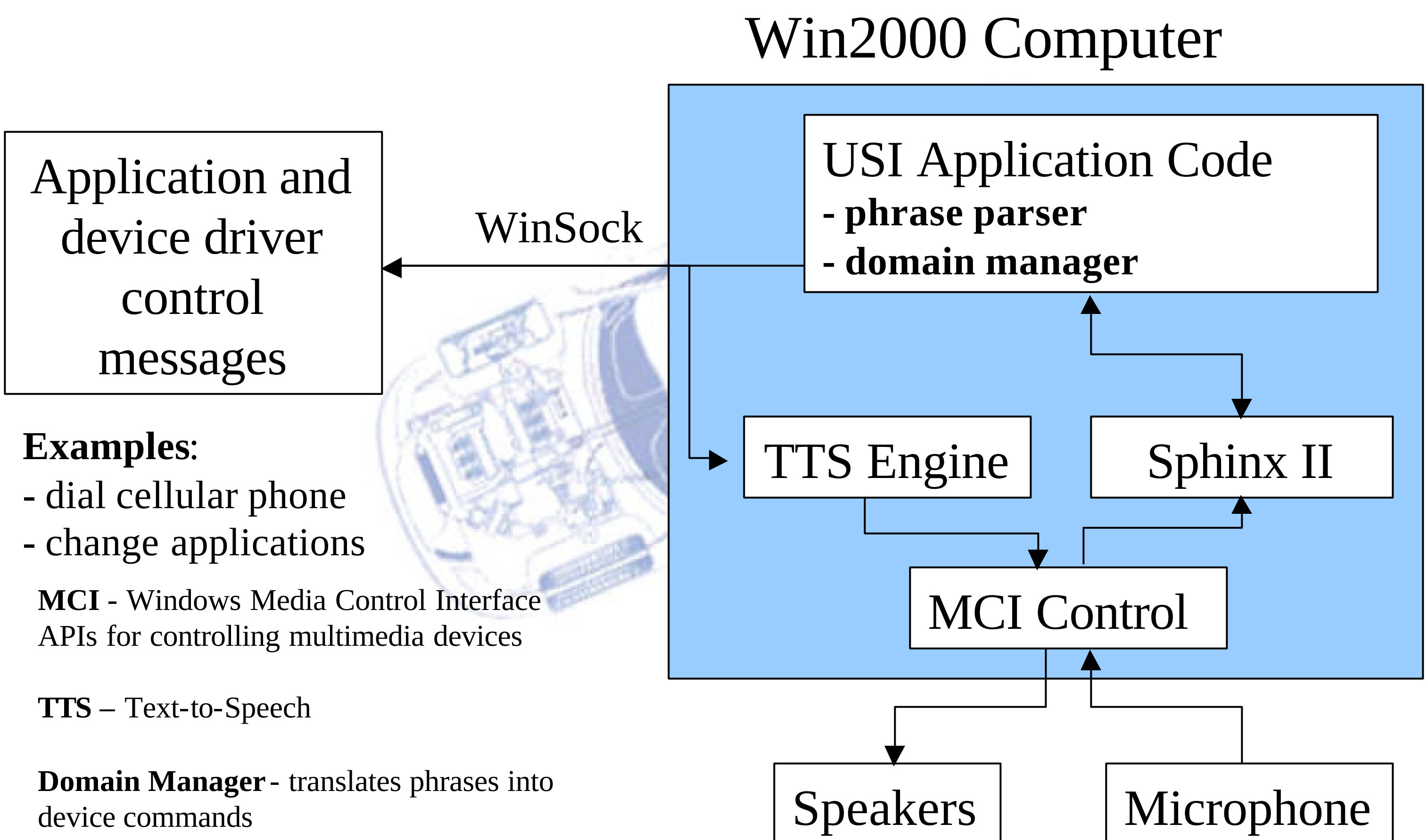
- A framework for developing structured speech dialogues for a smooth and accurate speech interface
- Incorporates
  - Format for utterances
  - Commands to ask for help and correct mistakes
  - Synthesized speech feedback to enhance usability
- Parses output from existing speech recognition engines; current implementation uses CMU’s Sphinx II

USI demo - CMU Apartment Search Line: (412) 268-1185

USI: User Interaction

- Format for user commands:
    - **[slot]** is **[value]**
        - Slot:
        - "**Contact** is **John**, **Action** is **Call**"
        - "**Application** is **Todo**"
- Command to ask for help: "Now what?"
    - Elicits a spoken list of slots
        - User says "Now What?"
        - Text-to-speech responds "Application is what, Contact is what, call is what…"
    - Used as a value: elicits a spoken list of possible values for that slot
        - User says: "**Application** is **Now What**?"
        - Text-to-speech responds: "Application is web browser, to do, contacts, schedule"

USI Architecture



Examples:
- dial cellular phone
- change applications

MCI - Windows Media Control Interface APIs for controlling multimedia devices

TTS – Text-to-Speech

Domain Manager - translates phrases into device commands
e.g. Phrase “*Application is to do*” is translated into a command for the LCD driver to display the to do list

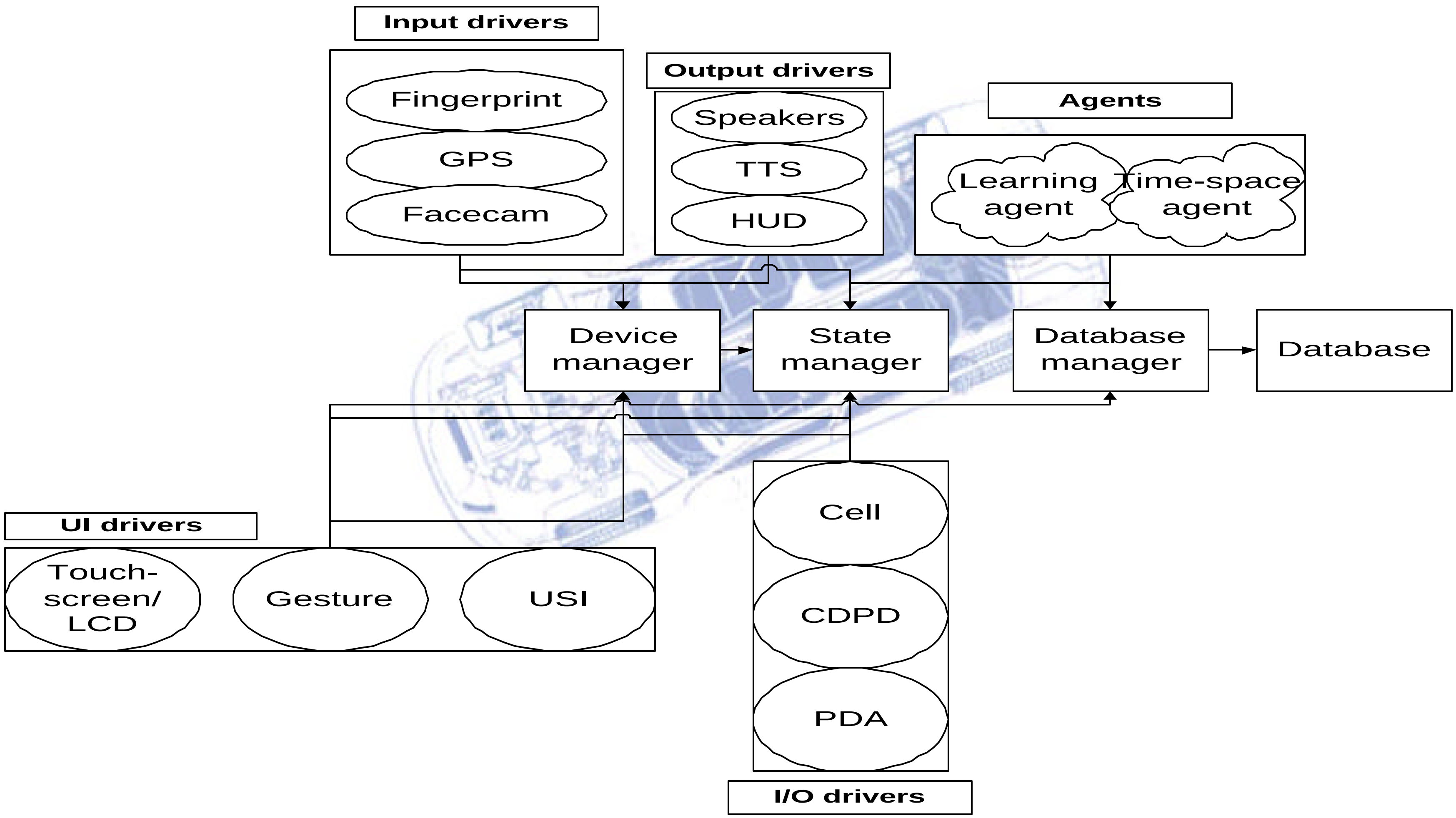
Software Architecture

- Allows connected software to be written in any language, and run on any platform
- Uses a publish/subscribe model
  - Connects together information providers and consumers by delivering events from sources to interested users
  - Programs can publish state information
  - Other programs can subscribe to published states; they receive notifications as states are modified
- All components in the system communicate using TCP/IP

Software Architecture

- **State manager**
  - Stores and manages frequently updated information
    - GPS coordinates, currently displayed alert, current user
  - Devices can query the value of a state or subscribe to changes in a state
- **Device manager**
  - Arbitrates access to resources
  - Queues and passes messages between devices
- **Database manager**
  - Wraps around MS SQL Server
  - Device/platform independent database interface; allows software on non-Windows platforms to access MS SQL Server without proprietary protocol programming
- **Connected to these are separate programs which implement a particular input, output, or input and output task of the Companion**

Architecture Diagram



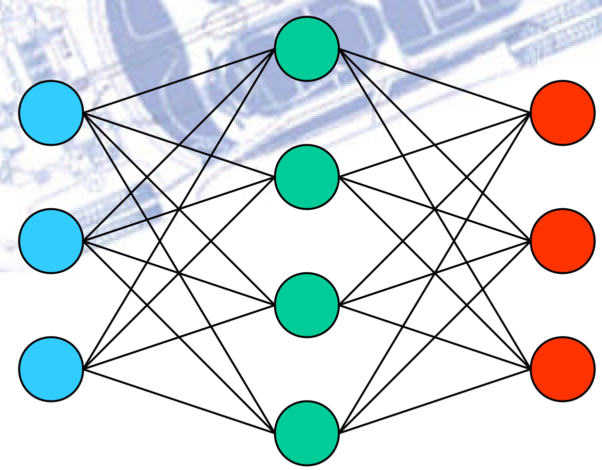
Agents

- Agents act proactively on behalf of the user
  - Space/Time Agent notifies the driver when he is about to pass a location at which he is scheduled to perform a task
  - Learning Agent configures car settings such as seat position and temperature. It learns a driver’s preferences by observing the actions he takes and the actions’ context using a neural network.

Learning Agent

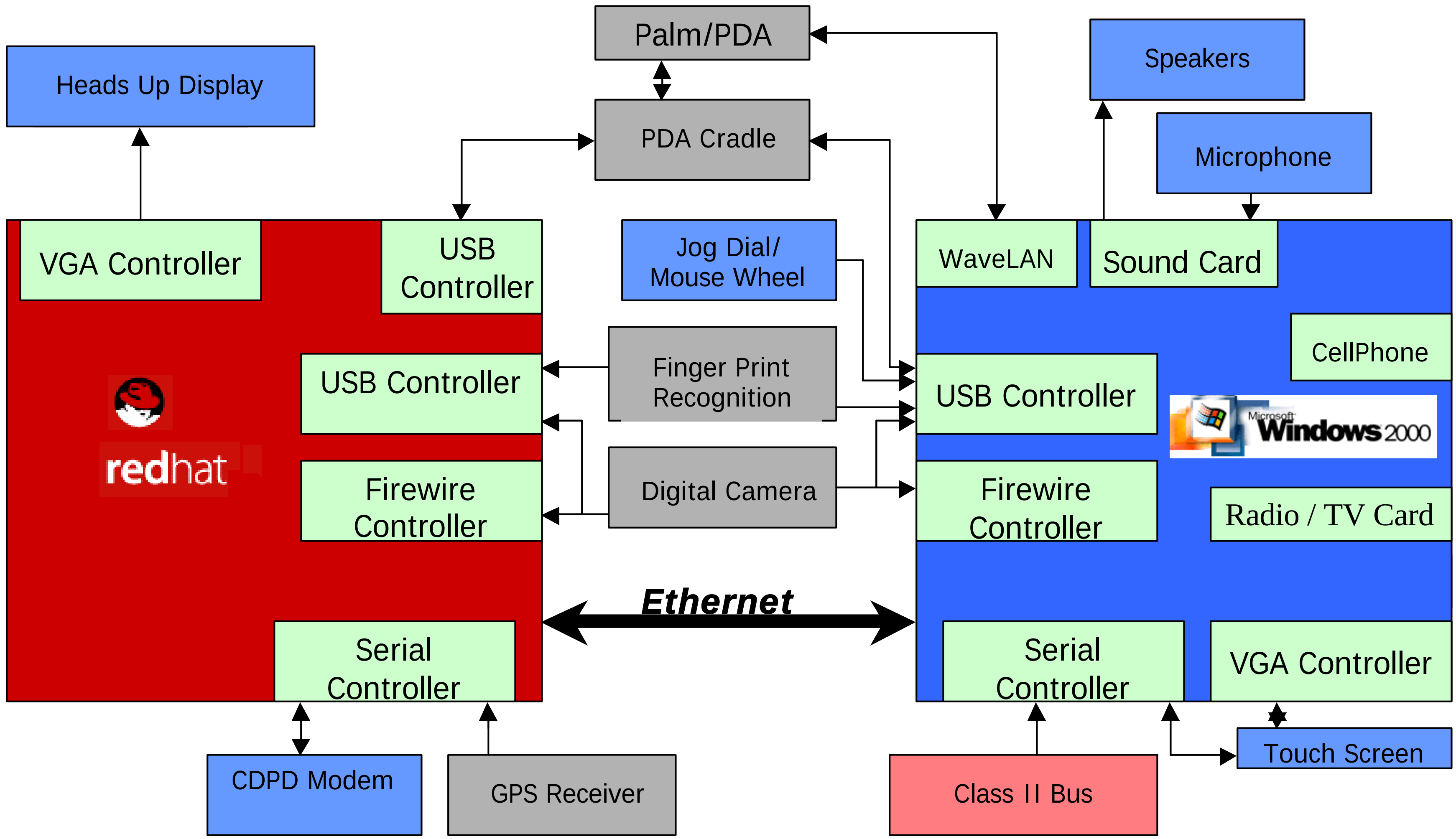
- **Supervised Learning:**
  - Neural Network Architecture
  - Uses estimation and observation-based prediction on actual outcomes to adjust link weights

Input Layer
- •Physical Settings
- •Car Internal State
- •Onboard Sensors
- •Interaction
- •Space-Time
- •Context
- •Previous Output



Output Layer
- •Physical Settings
- •Car Internal State
- •Interaction

Hardware Architecture



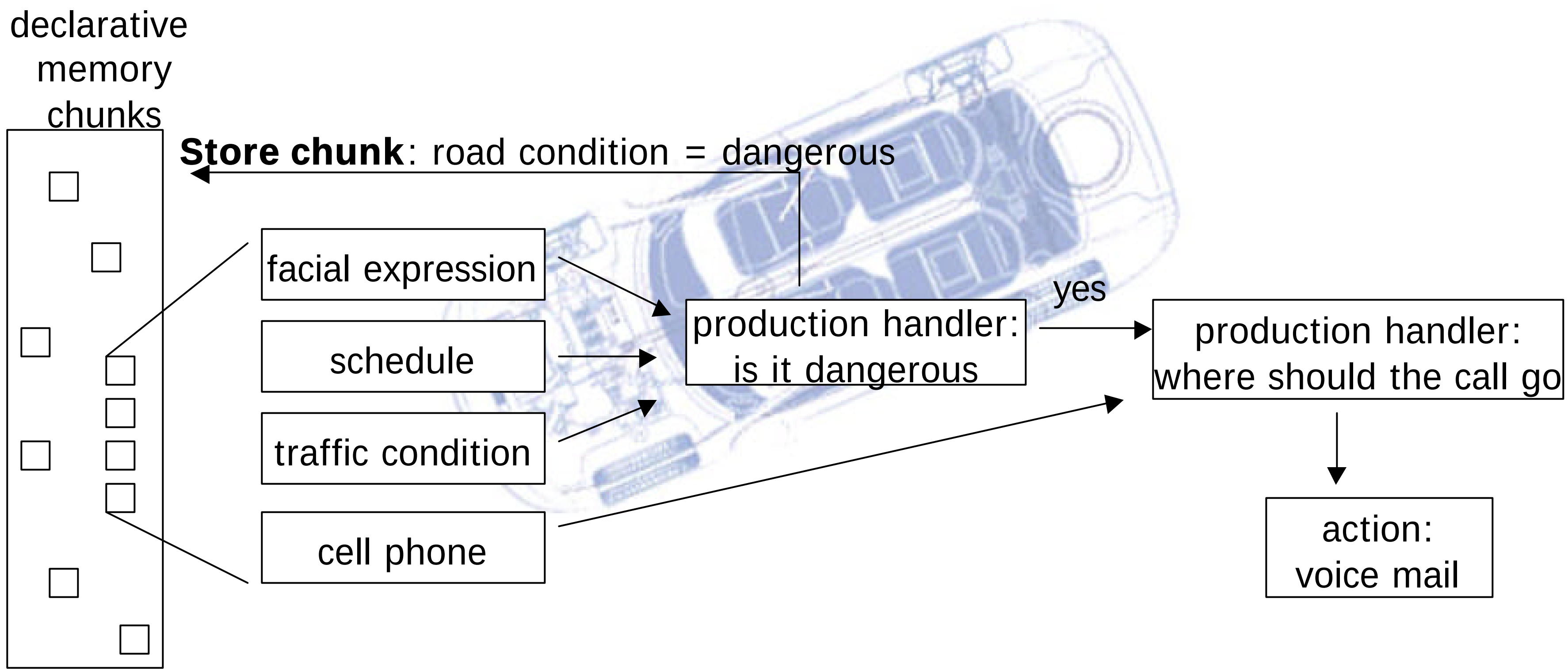
Cognitive Model for Context Aware Computing

- Create contextual car driver interface model that includes driver's cognitive state
- Driver's cognitive state includes
  - current goal
  - approach used to fulfill their intention
  - working memory (information needed for working on current tasks)
  - elements they perceive in their environment
- Extend CMU's ACT-R, which models the cognitive processes generating user intent.

ACT-R Components

- Declarative memory: collection of declarative knowledge elements stored in memory in the form slot = value
- Elements of declarative knowledge are called chunks
  - "The current driver is Tom"
  - "road condition is dangerous"
- Production rule: an "if / then" rule which fires a particular action if certain declarative memory chunks are matched
- Production handler: part of a program which executes production rules
- Production system: a system which uses production rules in conjunction with declarative memory to perform actions

Augmenting ACT-R to Contextual Driver Interface



If <cell phone = ringing> & <road condition = dangerous>
Then <action = send call to voice mail>

Conclusions

- Designed and implemented contextual car driver interface
  - Created an effective multimodal interface integrating many input and output modalities
  - Demonstrated the viability of included technologies, such as FaceCam/GestureCam and USI
  - Created a platform for research in context aware computing and smart spaces
- Designed and implemented several smart agents
- Created a cognitive model for a context aware software architecture
- Developed a multi-computer architecture as the platform for our prototype, integrated with the minivan

Effects of Multitasking