Exhibit E-Maes
Invalidity Chart for U.S. Patent No. 9,495,957 in view of U.S. Patent No. 6,964,023 ("Maes")

U.S. Patent No. 6,964,023 ("Maes") was published September 26, 2002, and issued November 8, 2005. Maes qualifies as prior art to U.S. Patent No. 9,495,957 (the "'957 Patent") at least under one or more applicable provisions of 35 U.S.C. §§ 102(a), (b), (e), and/or (g). Maes discloses and/or renders obvious one or more of claims 7 and 8 of the '957 Patent alone and/or in combination with other references, as set forth in the chart below. To the extent Maes does not disclose one or more limitations of the claims, such limitations would have been inherent and/or it would have been obvious to combine the disclosures of Maes with the knowledge of one of ordinary skill in the art and/or with one or more other references, including as identified herein and/or in the Cover Pleading, for the reasons identified herein and/or in the Cover Pleading, to render the claims at-issue in the '957 Patent invalid.

Defendant Bank of America, N.A. ("Bank") incorporates in this chart all applicable qualifications, clarifications, and other statements made in Bank's Invalidity Contentions. This invalidity claim chart is based on Bank's present understanding of claims 7, 8, and Plaintiff Dialect, LLC ("Dialect") apparent construction of the claims, as set forth in Dialect's Infringement Contentions. These invalidity assertions are not an admission that Bank concedes or acquiesces to any claim construction(s) implied or suggested by Dialect the associated infringement claim charts. Nor is Bank asserting any claim construction positions through these charts, including whether the preamble is a limitation. Bank also does not concede or acquiesce that any asserted claim satisfies the requirements of 35 U.S.C. §§ 112 or 101, and submits these invalidity contentions only to the extent Dialect's assertions may be understood.

The excerpts cited herein are exemplary. For any claim limitation, Bank may rely on excerpts cited for any other limitation and/or additional excerpts not set forth fully herein to the extent necessary to provide a more comprehensive explanation for a reference's disclosure of a limitation. Where an excerpt refers to or discusses a figure or figure items, that figure and any additional descriptions of that figure should be understood to be incorporated by reference as if set forth fully herein. To the extent any cell lacks citations to the charted reference(s), this should not be taken as an admission that the reference(s) do(es) not disclose the corresponding limitation but rather indicates that Bank does not presently have sufficient information regarding the charted reference(s) based on Bank's present understanding of the claim limitation.

These invalidity contentions are not an admission by Bank that the accused products or components, including any current or past version of the products or components, are covered by, or infringe the asserted claims, particularly when these claims are properly construed and applied.

1. Claim 7

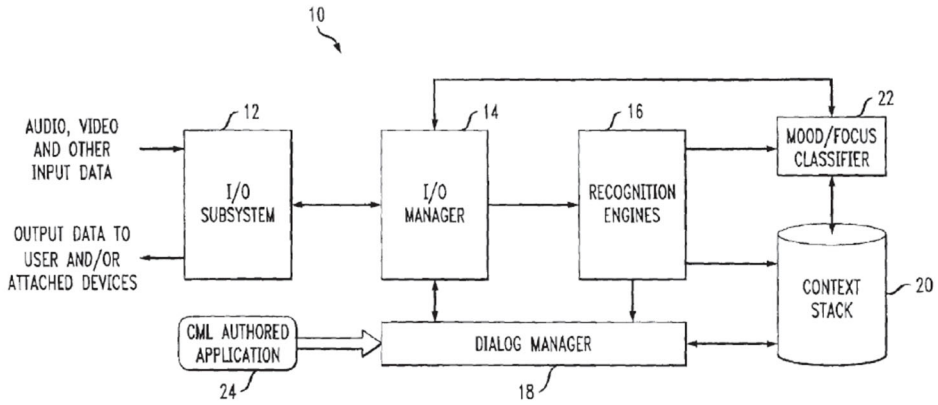
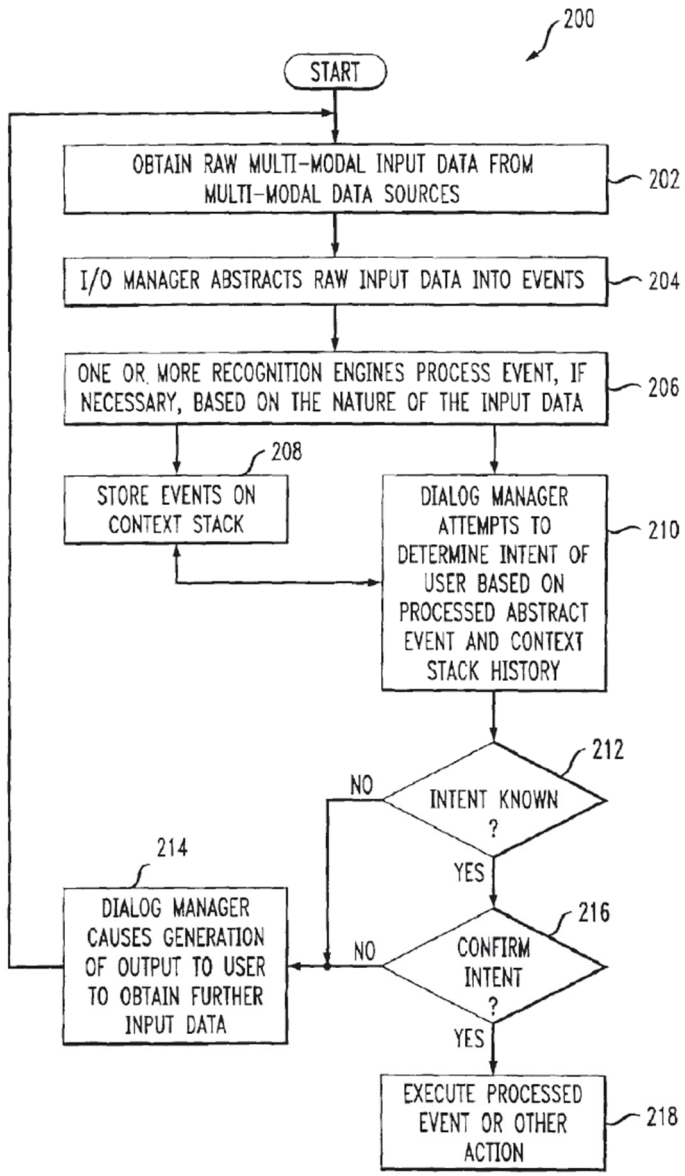
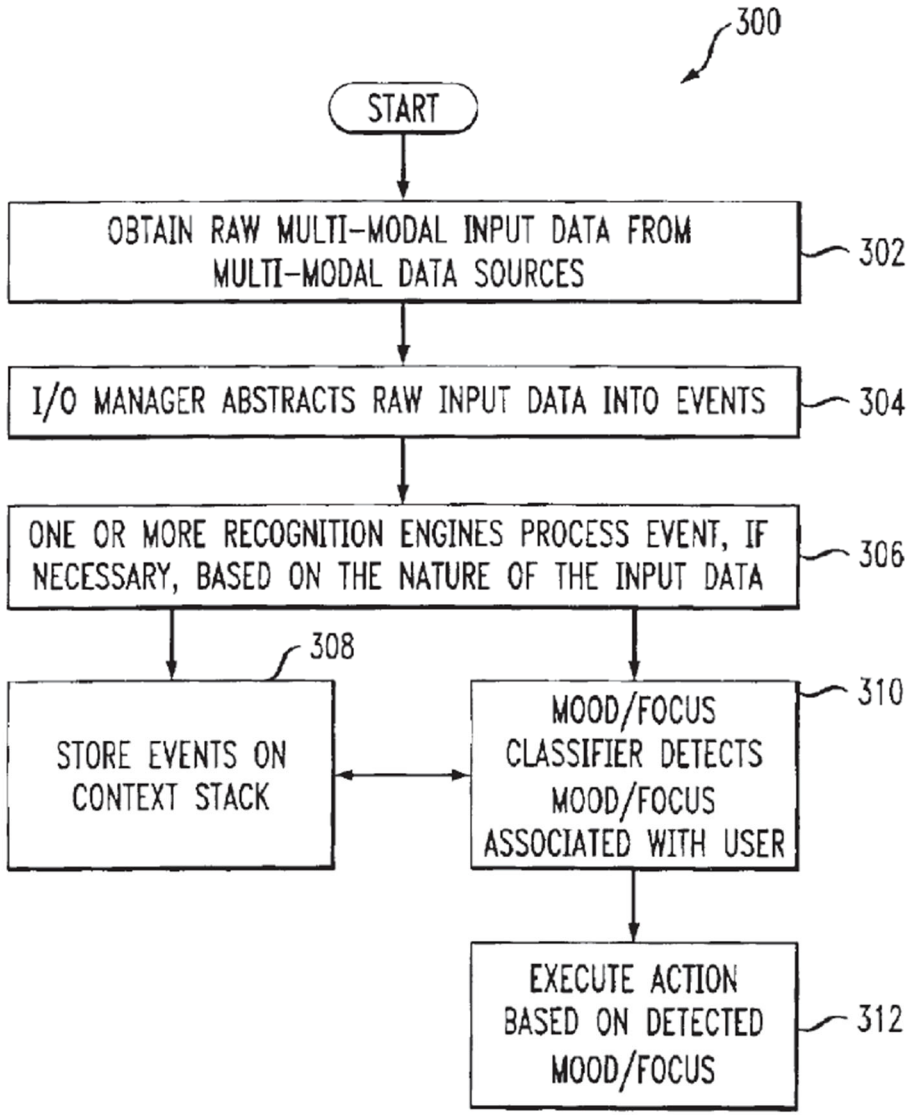
Claim 7 of the '957 Patent	Exemplary Citations to Maes
[p] A computer-implemented method of processing a natural language utterance, the method comprising:	To the extent the preamble is limiting, Maes discloses a "computer-implemented method of processing a natural language utterance[.]" *E.g.*, Maes at Abstract: Systems and methods are provided for performing focus detection, referential ambiguity resolution and mood classification in accordance with multi-modal input data, in varying operating conditions, in order to provide an effective conversational computing environment for one or more users. Maes at Fig. 1: *FIG. 1*  Maes at Fig. 2:

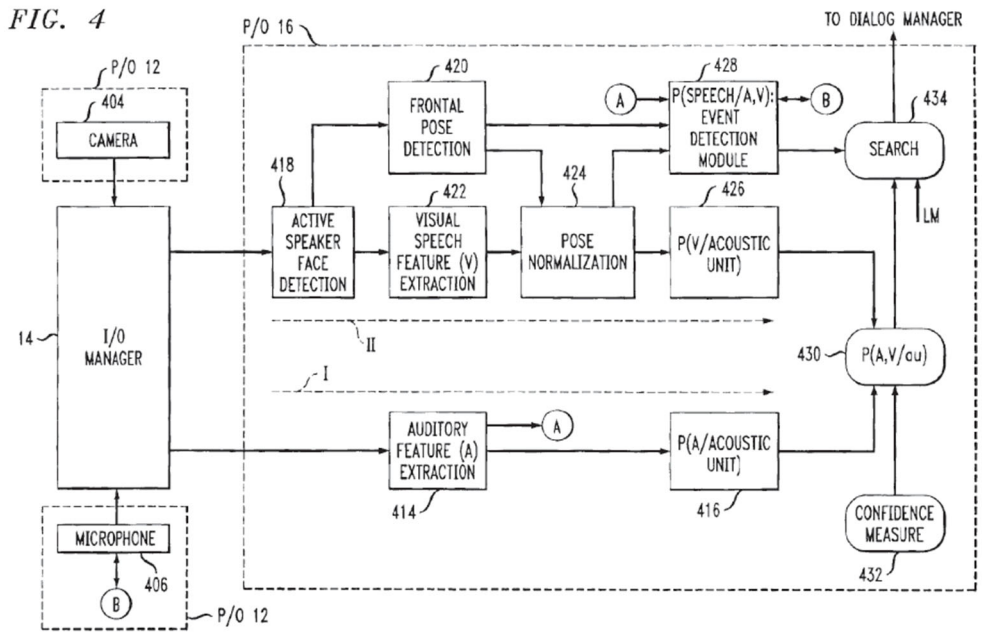
FIG. 2

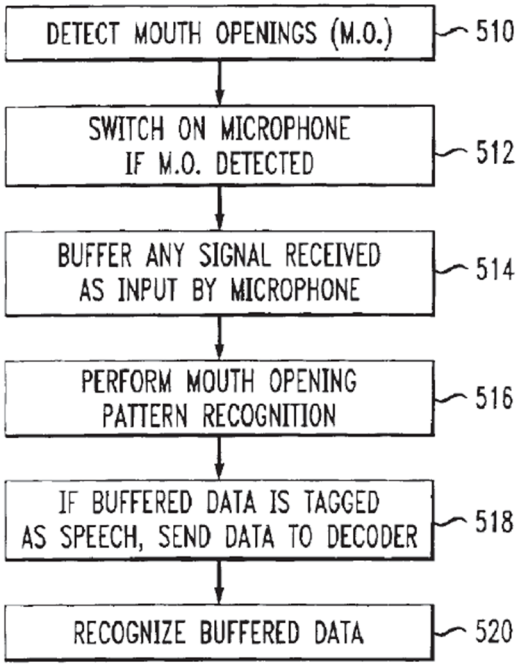

200
START
OBTAIN RAW MULTI-MODAL INPUT DATA FROM MULTI-MODAL DATA SOURCES
202
I/O MANAGER ABSTRACTS RAW INPUT DATA INTO EVENTS
204
ONE OR MORE RECOGNITION ENGINES PROCESS EVENT, IF NECESSARY, BASED ON THE NATURE OF THE INPUT DATA
206
208
STORE EVENTS ON CONTEXT STACK
DIALOG MANAGER ATTEMPTS TO DETERMINE INTENT OF USER BASED ON PROCESSED ABSTRACT EVENT AND CONTEXT STACK HISTORY
210
212
NO
INTENT KNOWN ?
YES
214
DIALOG MANAGER CAUSES GENERATION OF OUTPUT TO USER TO OBTAIN FURTHER INPUT DATA
216
NO
CONFIRM INTENT ?
YES
EXECUTE PROCESSED EVENT OR OTHER ACTION
218

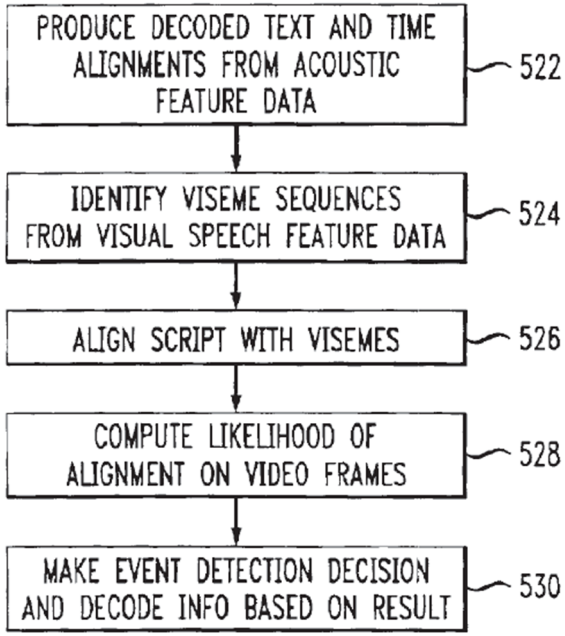
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Maes at Fig. 3:

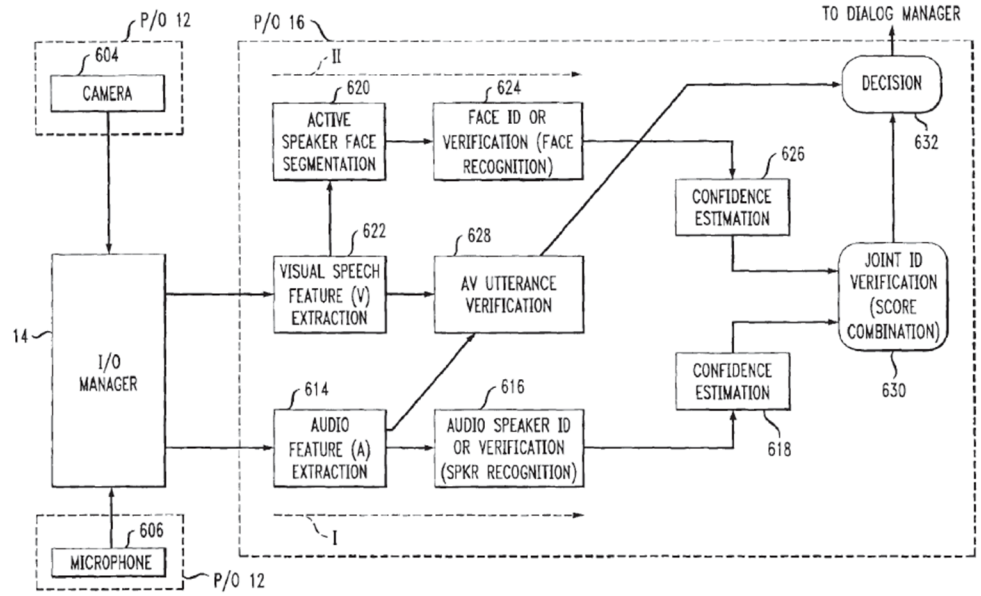
FIG. 3

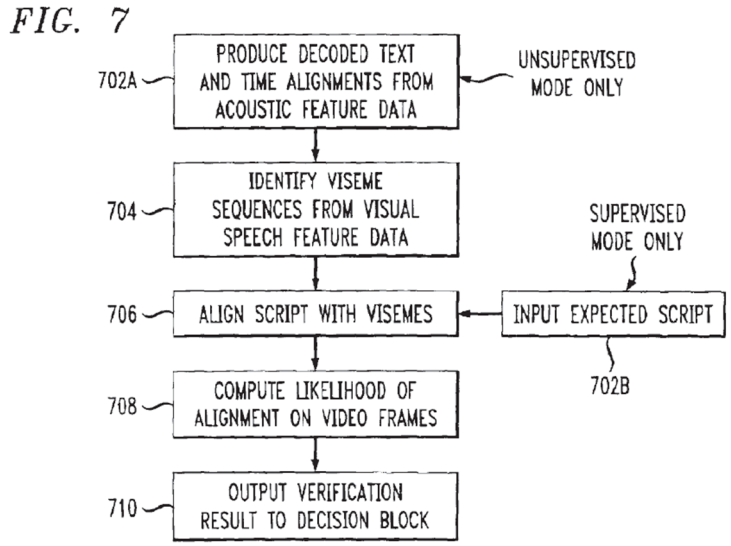

300
START
OBTAIN RAW MULTI-MODAL INPUT DATA FROM MULTI-MODAL DATA SOURCES
302
I/O MANAGER ABSTRACTS RAW INPUT DATA INTO EVENTS
304
ONE OR MORE RECOGNITION ENGINES PROCESS EVENT, IF NECESSARY, BASED ON THE NATURE OF THE INPUT DATA
306
308
STORE EVENTS ON CONTEXT STACK
MOOD/FOCUS CLASSIFIER DETECTS MOOD/FOCUS ASSOCIATED WITH USER
310
EXECUTE ACTION BASED ON DETECTED MOOD/FOCUS
312

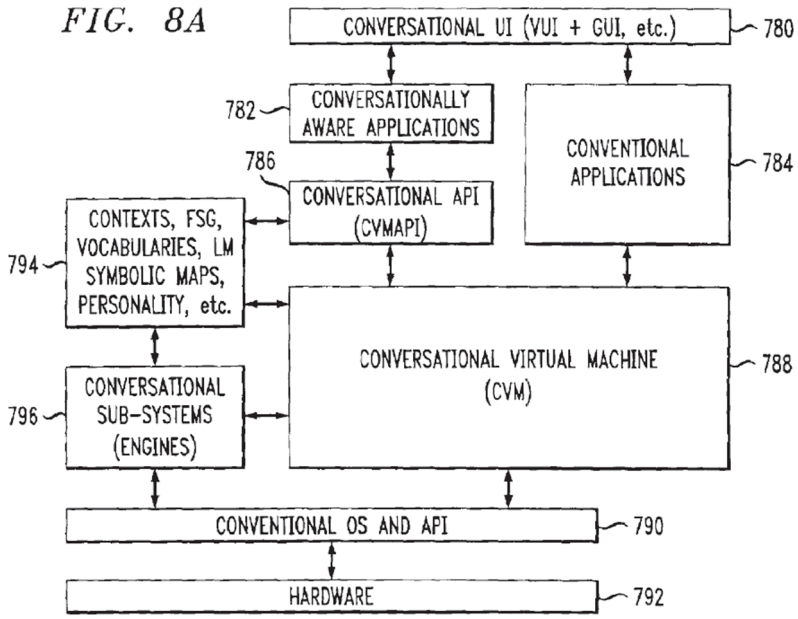
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Maes at Fig. 4:  Maes at Fig. 5C:

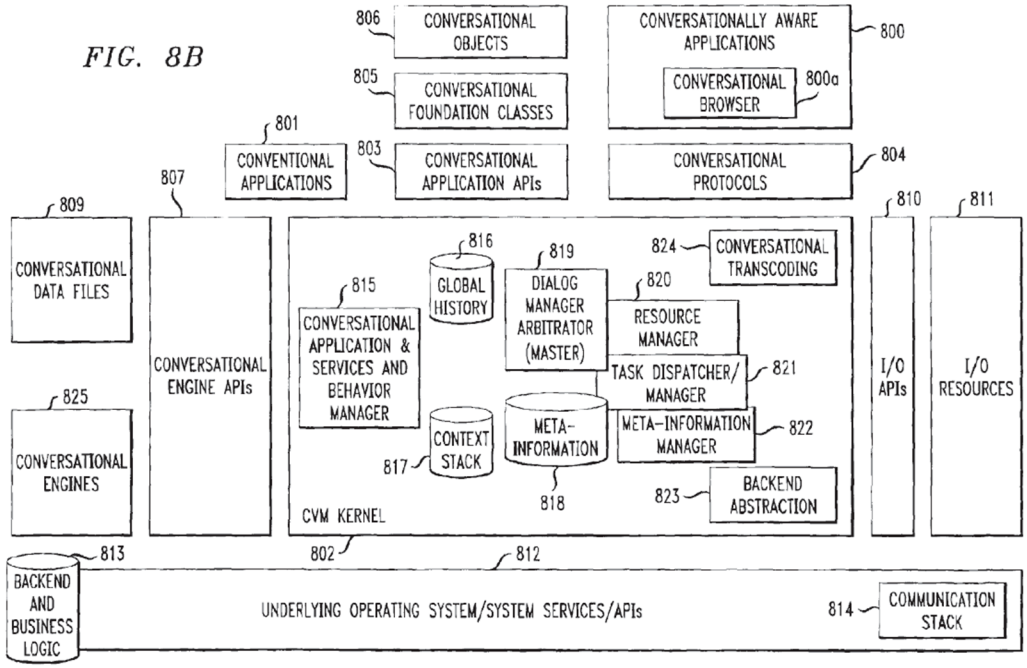
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	FIG. 5C  Maes at 5D:

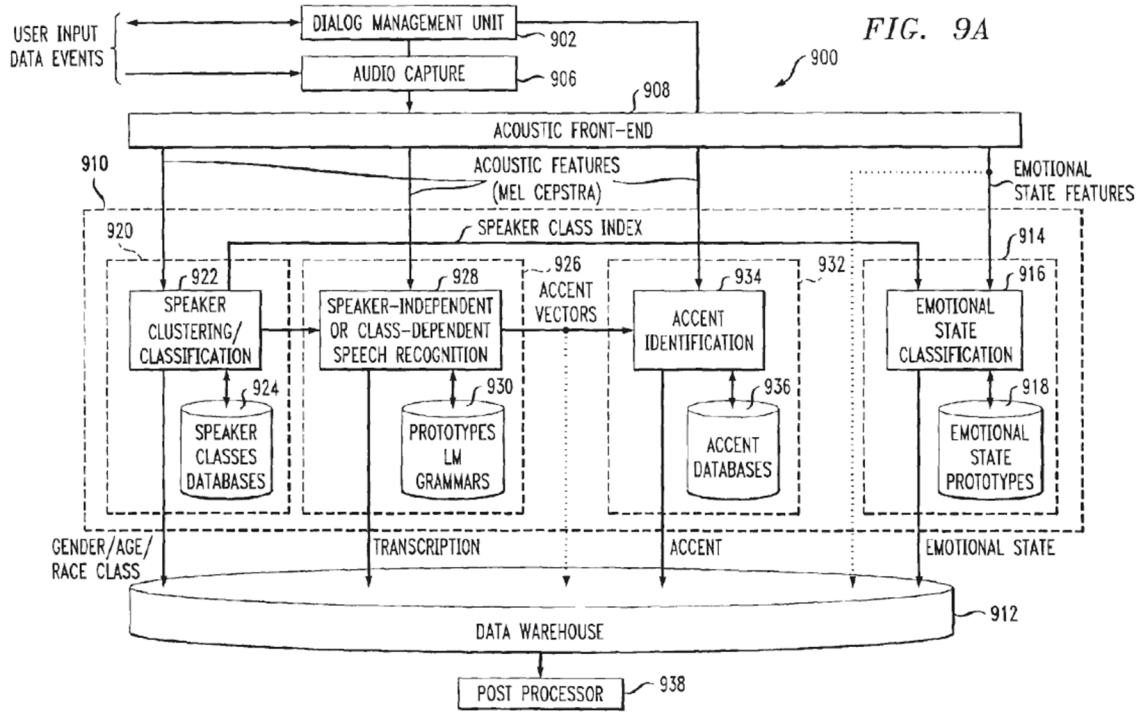
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	FIG. 5D  Maes at Fig. 6:

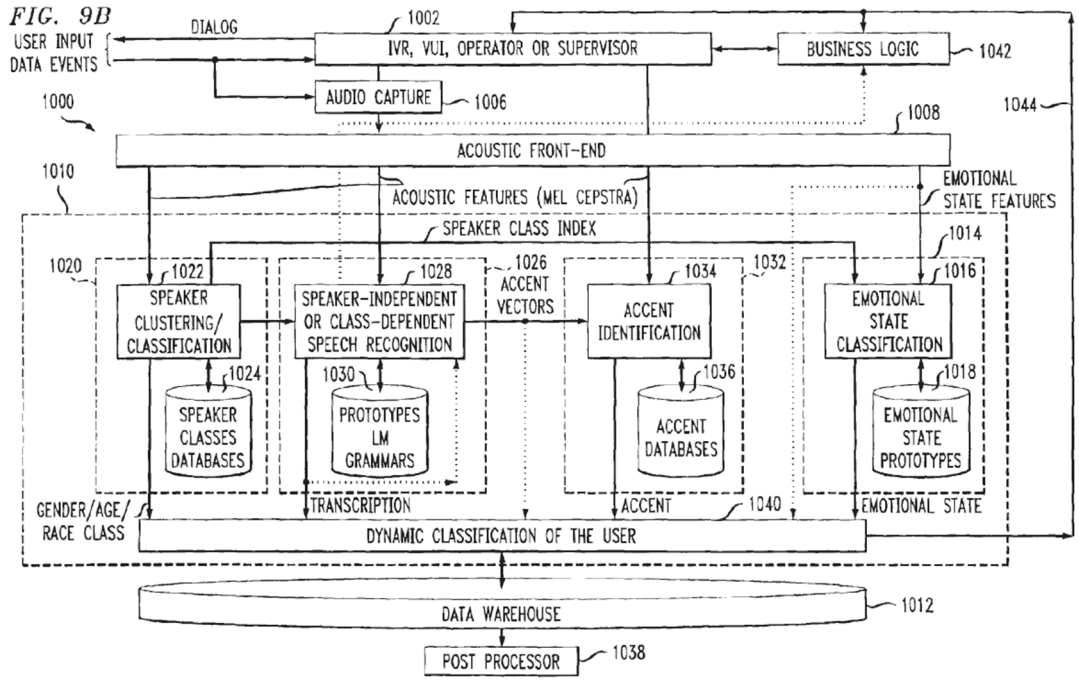
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	FIG. 6  Maes at Fig. 7:

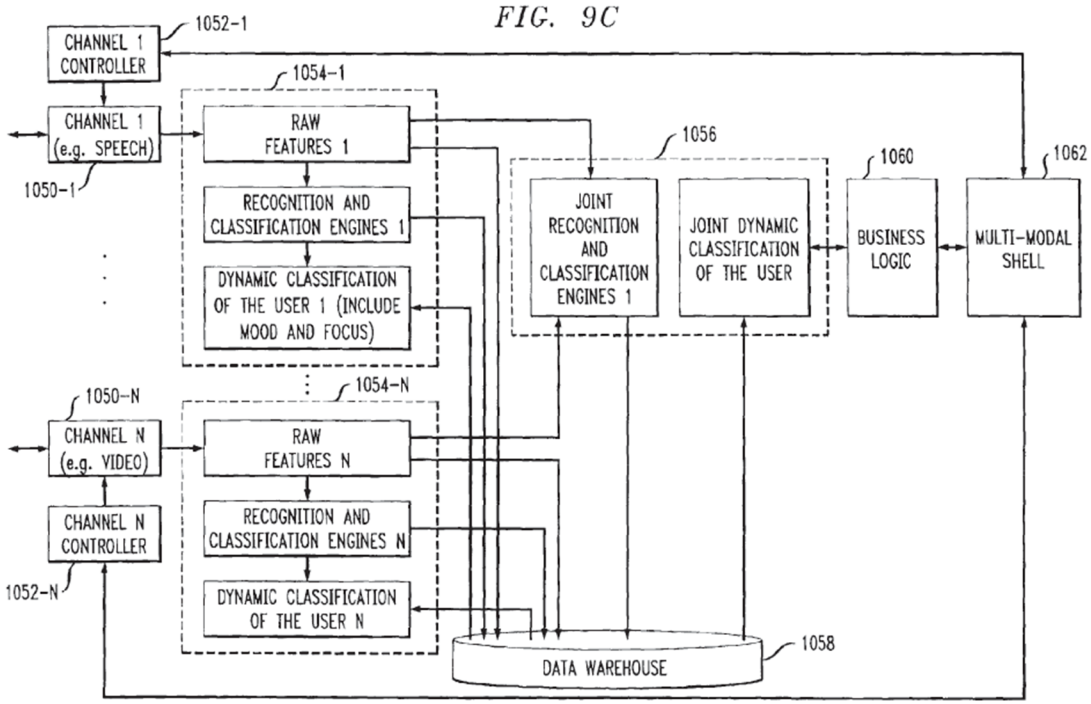
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 8A:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig 8B:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 9A:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 9B:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 9C:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	FIG. 9C  Maes at 3:66-4:21 Referring initially to FIG. 1, a block diagram illustrates a multi-modal conversational computing system according to an embodiment of the present invention. As shown, the multi-modal conversational computing system 10 comprises an input/output (I/O) subsystem 12, an I/O manager module 14, one or more recognition engines 16, a dialog manager module 18, a context stack 20 and a mood/focus classifier 22. Generally, the multi-modal conversational computing system 10 of the present invention receives multi-modal input in the form of audio input data, video input data, as well as other types of input

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	data (in accordance with the I/O subsystem 12), processes the multi-modal data (in accordance with the I/O manager 14), and performs various recognition tasks (e.g., speech recognition, speaker recognition, gesture recognition, lip reading, face recognition, etc., in accordance with the recognition engines 16), if necessary, using this processed data. The results of the recognition tasks and/or the processed data, itself, is then used to perform one or more conversational computing tasks, e.g., focus detection, referential ambiguity resolution, and mood classification (in accordance with the dialog manager 18, the context stack 20 and/or the classifier 22), as will be explained in detail below. Maes at 4:62-5:5: Thus, the system 10 must be able to perform user reference resolution, e.g., the system may receive the spoken utterance, “call my office,” but unless the system can resolve which occupant made this statement, it will not know which office phone number to direct an associated cellular telephone to call. The system 10 therefore performs referential ambiguity resolution with respect to multiple users by taking both audio input data and image data input and processing it to make a user resolution determination. This may include detecting speech activity and/or the identity of the user based on both audio and image cues. Maes at 5:7-15: Similarly, a user may say to the system, “turn that off,” but without device reference resolution, the system would not know which associated device to direct to be turned off. The system 10 therefore performs referential ambiguity resolution with respect to multiple devices by taking both audio input data and image data input and processing it to make a device resolution determination. This may include detecting the speaker's head pose using gross spatial resolution of the direction being addressed, or body pose (e.g., pointing). Maes at 5:15-19:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	This may also include disambiguating an I/O (input/output) event generated previously and stored in a context manager/history stack (e.g., if a beeper rang and the user asked “turn it off,” the term “it” can be disambiguated). Techniques for accomplishing this will be explained below. Maes at 5:57-6:20: First, in step 202, raw multi-modal input data is obtained from multi-modal data sources associated with the system. In terms of the computing system 10 in FIG. 1, such sources are represented by I/O subsystem 12. As mentioned above, the data input portion of the subsystem may comprise one or more cameras or sensors for capturing video input data representing the environment in which the system (or, at least, the I/O subsystem) is deployed. The cameras/sensors may be capable of capturing not only visible image data (images in the visible electromagnetic spectrum), but also IR (near, mid and/or far field IR video) and/or RF image data. Of course, in systems with more than one camera, different mixes of cameras/sensors may be employed, e.g., system having one or more video cameras, one or more IR sensors and/or one or more RF sensors. In addition to the one or more cameras, the I/O subsystem 12 may comprise one or more microphones for capturing audio input data from the environment in which the system is deployed. Further, the I/O subsystem may also include an analog-to-digital converter which converts the electrical signal generated by a microphone into a digital signal representative of speech uttered or other sounds that are captured. Further, the subsystem may sample the speech signal and partition the signal into overlapping frames so that each frame is discretely processed by the remainder of the system. Thus, referring to the vehicle example above, it is to be understood that the cameras and microphones may be strategically placed throughout the vehicle in order to attempt to fully capture all visual activity and audio activity that may be necessary for the system to make ambiguity resolution determinations. Maes at 6:21-38:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Still further, the I/O subsystem 12 may also comprise other typical input devices for obtaining user input, e.g., GUI-based devices such as a keyboard, a mouse, etc., and/or other devices such as a stylus and digitizer pad for capturing electronic handwriting, etc. It is to be understood that one of ordinary skill in the art will realize other user interfaces and devices that may be included for capturing user activity. Next, in step 204, the raw multi-modal input data is abstracted into one or more events. In terms of the computing system 10 in FIG. 1, the data abstraction is performed by the I/O manager 14. The I/O manager receives the raw multi-modal data and abstracts the data into a form that represents one or more events, e.g., a spoken utterance, a visual gesture, etc. As is known, a data abstraction operation may involve generalizing details associated with all or portions of the input data so as to yield a more generalized representation of the data for use in further operations. Maes at 6:39-7:20: In step 206, the abstracted data or event is then sent by the I/O manager 14 to one or more recognition engines 16 in order to have the event recognized, if necessary. That is, depending on the nature of the event, one or more recognition engines may be used to recognize the event. For example, if the event is some form of spoken utterance wherein the microphone picks up the audible portion of the utterance and a camera picks up the visual portion (e.g., lip movement) of the utterance, the event may be sent to an audio-visual speech recognition engine to have the utterance recognized using both the audio input and the video input associated with the speech. Alternatively, or in addition, the event may be sent to an audio-visual speaker recognition engine to have the speaker of the utterance identified, verified and/or authenticated. Also, both speech recognition and speaker recognition can be combined on the same utterance. If the event is some form of user gesture picked up by a camera, the event may be sent to a gesture recognition engine for recognition. Again, depending on the types of user interfaces provided by the system, the event may comprise handwritten input provided by the user such that one of the recognition engines may be a handwriting recognition engine. In the case of more typical GUI-

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	based input (e.g., keyboard, mouse, etc.), the data may not necessarily need to be recognized since the data is already identifiable without recognition operations. An audio-visual speech recognition module that may be employed as one of the recognition engines 16 is disclosed in U.S. patent application identified as Ser. No. 09/369,707, filed on Aug. 6, 1999 and entitled “Methods and Apparatus for Audio-visual Speech Detection and Recognition,” the disclosure of which is incorporated by reference herein. A description of such an audio-visual speech recognition system will be provided below. An audio-visual speaker recognition module that may be employed as one of the recognition engines 16 is disclosed in U.S. patent application identified as Ser. No. 09/369,706, filed on Aug. 6, 1999 and entitled “Methods And Apparatus for Audio-Visual Speaker Recognition and Utterance Verification,” the disclosure of which is incorporated by reference herein. A description of such an audio-visual speaker recognition system will be provided below. It is to be appreciated that gesture recognition (e.g., body, arms and/or hand movement, etc., that a user employs to passively or actively give instruction to the system) and focus recognition (e.g., direction of face and eyes of a user) may be performed using the recognition modules described in the above-referenced patent applications. Maes at 7:23-33: It is to be appreciated that two, more or even all of the input modes described herein may be synchronized via the techniques disclosed in U.S. patent application identified as Ser. No. 09/507,526 filed on Feb. 18, 2000 and entitled “Systems and Method for Synchronizing Multi-modal Interactions,” which claims priority to U.S. provisional patent application identified as U.S. Ser. No. 60/128,081 filed on Apr. 7, 1999 and U.S. provisional patent application identified by Ser. No. 60/158,777 filed on Oct. 12, 1999, the disclosures of which are incorporated by reference herein. Maes at 7:33-45: In step 208, the recognized events, as well as the events that do not need to be recognized, are stored in a storage unit referred to as the context stack 20. The context stack is used to create a

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	history of interaction between the user and the system so as to assist the dialog manager 18 in making referential ambiguity resolution determinations when determining the user's intent. Next, in step 210, the system 10 attempts to determine the user intent based on the current event and the historical interaction information stored in the context stack and then determine and execute one or more application programs that effectuate the user's intention and/or react to the user activity. Maes at 7:60-8:65: Thus, the dialog manager must first determine the user's intent based on the current event and, if available, the historical information (e.g., past events) stored in the context stack. For instance, returning to the vehicle example, the user may say “turn it on,” while pointing at the vehicle radio. The dialog manager would therefore receive the results of the recognized events associated with the spoken utterance “turn it on” and the gesture of pointing to the radio. Based on these events, the dialog manager does a search of the existing applications, transactions or “dialogs,” or portions thereof, with which such an utterance and gesture could be associated. Accordingly, as shown in FIG. 1, the dialog manager 18 determines the appropriate CML-authored application 24. The application may be stored on the system 10 or accessed (e.g., downloaded) from some remote location. If the dialog manager determines with some predetermined degree of confidence that the application it selects is the one which will effectuate the users desire, the dialog manager executes the next step of the multi-modal dialog (e.g., prompt or display for missing, ambiguous or confusing information, asks for confirmation or launches the execution of an action associated to a fully understood multi-modal request from the user) of that application based on the multi-modal input. That is, the dialog manager selects the appropriate device (e.g., radio) activation routine and instructs the I/O manager to output a command to activate the radio. The predetermined degree of confidence may be that at least two input parameters or variables of the application are satisfied or provided by the received events. Of course, depending on the application, other levels of confidence and algorithms may be established as, for example, described in K. A. Papineni, S. Roukos, R. T. Ward, “Free-flow dialog management using forms,” Proc. Eurospeech, Budapest, 1999; and K. Davies et al., “The IBM conversational telephony system for financial applications,” Proc. Eurospeech, Budapest, 1999, the disclosures of which are incorporated by reference herein.

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Consider the case where the user first says “turn it on,” and then a few seconds later points to the radio. The dialog manager would first try to determine user intent based solely on the “turn it on” command. However, since there are likely other devices in the vehicle that could be turned on, the system would likely not be able to determine with a sufficient degree of confidence what the user was referring to. However, this recognized spoken utterance event is stored on the context stack. Then, when the recognized gesture event (e.g., pointing to the radio) is received, the dialog manager takes this event and the previous spoken utterance event stored on the context stack and makes a determination that the user intended to have the radio turned on. Consider the case where the user says “turn it on,” but makes no gesture and provides no other utterance. In this case, assume that the dialog manager does not have enough input to determine the user intent (step 212 in FIG. 2) and thus implement the command. The dialog manager, in step 214, then causes the generation of an output to the user requesting further input data so that the user's intent can be disambiguated. This may be accomplished by the dialog manager instructing the I/O manager to have the I/O subsystem output a request for clarification. In one embodiment, the I/O subsystem 12 may comprise a text-to-speech (TTS) engine and one or more output speakers. The dialog manager then generates a predetermined question such as “what device do you want to have turned on?” which the TTS engine converts to a synthesized utterance that is audibly output by the speaker to the user. The user, hearing the query, could then point to the radio or say “the radio” thereby providing the dialog manager with the additional input data to disambiguate his request. That is, with reference to FIG. 2, the system 10 obtains the raw input data, again in step 202, and the process 200 iterates based on the new data. Such iteration can continue as long as necessary for the dialog manager to determine the user's intent. Maes at 8:66-9:28: The dialog manager 18 may also seek confirmation in step 216 from the user in the same manner as the request for more information (step 214) before executing the processed event, dispatching a task and/or executing some other action in step 218 (e.g., causing the radio to be turned on). For example, the system may output “do you want the radio turned on?” To which the user may

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	respond “yes.” The system then causes the radio to be turned on. Further, the dialog manager 18 may store information it generates and/or obtains during the processing of a current event on the context stack 20 for use in making resolution or other determinations at some later time. Of course, it is to be understood that the above example is a simple example of device ambiguity resolution. As mentioned, the system 10 can also make user ambiguity resolution determinations, e.g., in a multiple user environment, someone says “dial my office.” Given the explanation above, one of ordinary skill will appreciate how the system 10 could handle such a command in order to decide who among the multiple users made the request and then effectuate the order. Also, the output to the user to request further input may be made in any other number of ways and with any amount of interaction turns between the user and feedback from the system to the user. For example, the I/O subsystem 12 may include a GUI-based display whereby the request is made by the system in the form of a text message displayed on the screen of the display. One of ordinary skill in the art will appreciate many other output mechanisms for implementing the teachings herein. Maes at 10:8-19: The focus/mood classifier 22 receives either the events directly from the I/O manager 14 or, if necessary depending on the nature of the event, the classifier receives the recognized events from the one or more recognition engines 16. For instance, in the vehicle example, the focus/mood classifier may receive visual events indicating the position of the user's eyes and/or head as well as audio events indicating sounds the user may be making (e.g., snoring). Using these events, as well as past information stored on the context stack, the classifier makes the focus detection and/or mood classification determination. Results of such determinations may also be stored on the context stack. Maes at 10:20-29:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Then, in step 312, the classifier may cause the execution of some action depending on the resultant determination. For example, if the driver's attention is determined to be distracted, the I/O manager may be instructed by the classifier to output a warning message to the driver via the TTS system and the one or more output speakers. If the driver is determined to be tired due, for example, to his monitored body posture, the I/O manager may be instructed by the classifier to provide a warning message, adjust the temperature or radio volume in the vehicle, etc. Maes at 10:47-11:1: Referring now to FIG. 4, a block diagram illustrates a preferred embodiment of an audio-visual speech recognition module that may be employed as one of the recognition modules of FIG. 1 to perform speech recognition using multi-modal input data received in accordance with the invention. It is to be appreciated that such an audio-visual speech recognition module is disclosed in the above-referenced U.S. patent application identified as Ser. No. 09/369,707, filed on Aug. 6, 1999 and entitled “Methods and Apparatus for Audio-visual Speech Detection and Recognition.” A description of one of the embodiments of such an audio-visual speech recognition module for use in a preferred embodiment of the multi-modal conversational computing system of the invention is provided below in this section. However, it is to be appreciated that other mechanisms for performing speech recognition may be employed. This particular illustrative embodiment, as will be explained, depicts audio-visual recognition using a decision fusion approach. It is to be appreciated that one of the advantages that the audio-visual speech recognition module described herein provides is the ability to process arbitrary content video. Maes at 11:8-31: However, as will be explained in detail below, the module described herein is preferably able to process arbitrary content video which may not only contain faces but may also contain arbitrary background objects in a noisy environment. One example of arbitrary content video is in the context of broadcast news. Such video can possibly contain a newsperson speaking at a location

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	where there is arbitrary activity and noise in the background. In such a case, as will be explained, the module is able to locate and track a face and, more particularly, a mouth, to determine what is relevant visual information to be used in more accurately recognizing the accompanying speech provided by the speaker. The module is also able to continue to recognize when the speaker's face is not visible (audio only) or when the speech in inaudible (lip reading only). Thus, the module is capable of receiving real-time arbitrary content from a video camera 404 and microphone 406 via the I/O manager 14. It is to be understood that the camera and microphone are part of the I/O subsystem 12. While the video signals received from the camera 404 and the audio signals received from the microphone 406 are shown in FIG. 4 as not being compressed, they may be compressed and therefore need to be decompressed in accordance with the applied compression scheme. Maes at 11:32-54: It is to be understood that the video signal captured by the camera 404 can be of any particular type. As mentioned, the face and pose detection techniques may process images of any wavelength such as, e.g., visible and/or non-visible electromagnetic spectrum images. By way of example only, this may include infrared (IR) images (e.g., near, mid and far field IR video) and radio frequency (RF) images. Accordingly, the module may perform audio-visual speech detection and recognition techniques in poor lighting conditions, changing lighting conditions, or in environments without light. For example, the system may be installed in an automobile or some other form of vehicle and capable of capturing IR images so that improved speech recognition may be performed. Because video information (i.e., including visible and/or non-visible electromagnetic spectrum images) is used in the speech recognition process, the system is less susceptible to recognition errors due to noisy conditions, which significantly hamper conventional recognition systems that use only audio information. In addition, due to the methodologies for processing the visual information described herein, the module provides the capability to perform accurate LVCSR (large vocabulary continuous speech recognition). Maes at 11:64-12:28:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	The module includes an auditory feature extractor 414. The feature extractor 414 receives an audio or speech signal and, as is known in the art, extracts spectral features from the signal at regular intervals. The spectral features are in the form of acoustic feature vectors (signals) which are then passed on to a probability module 416. Before acoustic vectors are extracted, the speech signal may be sampled at a rate of 16 kilohertz (kHz). A frame may consist of a segment of speech having a 25 millisecond (msec) duration. In such an arrangement, the extraction process preferably produces 24 dimensional acoustic cepstral vectors via the process described below. Frames are advanced every 10 msec to obtain succeeding acoustic vectors. Note that other acoustic front-ends with other frame sizes and sampling rates/signal bandwidths can also be employed. First, in accordance with a preferred acoustic feature extraction process, magnitudes of discrete Fourier transforms of samples of speech data in a frame are considered in a logarithmically warped frequency scale. Next, these amplitude values themselves are transformed to a logarithmic scale. The latter two steps are motivated by a logarithmic sensitivity of human hearing to frequency and amplitude. Subsequently, a rotation in the form of discrete cosine transform is applied. One way to capture the dynamics is to use the delta (first-difference) and the delta-delta (second-order differences) information. An alternative way to capture dynamic information is to append a set of (e.g., four) preceding and succeeding vectors to the vector under consideration and then project the vector to a lower dimensional space, which is chosen to have the most discrimination. The latter procedure is known as Linear Discriminant Analysis (LDA) and is well known in the art. Maes at 12:29-48: After the acoustic feature vectors, denoted in FIG. 4. by the letter A, are extracted, the probability module labels the extracted vectors with one or more previously stored phonemes which, as is known in the art, are sub-phonetic or acoustic units of speech. The module may also work with lefemes, which are portions of phones in a given context. Each phoneme associated with one or more feature vectors has a probability associated therewith indicating the likelihood that it was that particular acoustic unit that was spoken. Thus, the probability module yields likelihood scores

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	for each considered phoneme in the form of the probability that, given a particular phoneme or acoustic unit (au), the acoustic unit represents the uttered speech characterized by one or more acoustic feature vectors A or, in other words, P(A|acoustic unit). It is to be appreciated that the processing performed in blocks 414 and 416 may be accomplished via any conventional acoustic information recognition system capable of extracting and labeling acoustic feature vectors, e.g., Lawrence Rabiner, Biing-Hwang Juang, “Fundamentals of Speech Recognition,” Prentice Hall, 1993. Maes at 12:49-60: Referring now to the video signal path II of FIG. 4, the methodologies of processing visual information will now be explained. The audio-visual speech recognition module (denoted in FIG. 4 as part of block 16 from FIG. 1) includes an active speaker face detection module 418. The active speaker face detection module 418 receives video input camera 404. It is to be appreciated that speaker face detection can also be performed directly in the compressed data domain and/or from audio and video information rather than just from video information. In any case, module 418 generally locates and tracks the speaker's face and facial features within the arbitrary video background. Maes at 12:62-13:15: The recognition module also preferably includes a frontal pose detection module 420. It is to be understood that the detection module 420 serves to determine whether a speaker in a video frame is in a frontal pose. This serves the function of reliably determining when someone is likely to be uttering or is likely to start uttering speech that is meant to be processed by the module, e.g., recognized by the module. This is the case at least when the speaker's face is visible from one of the cameras. When it is not, conventional speech recognition with, for example, silence detection, speech activity detection and/or noise compensation can be used. Thus, background noise is not recognized as though it were speech, and the starts of utterances are not mistakenly discarded. It is to be appreciated that not all speech acts performed within the hearing of the module are intended for the system. The user may not be speaking to the system, but to another person present or on the

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	telephone. Accordingly, the module implements a detection module such that the modality of vision is used in connection with the modality of speech to determine when to perform certain functions in auditory and visual speech recognition. Maes at 18:42-63: The extracted visual speech feature vectors are then normalized in block 424 with respect to the frontal pose estimates generated by the detection module 420. The normalized visual speech feature vectors are then provided to a probability module 426. Similar to the probability module 416 in the audio information path which labels the acoustic feature vectors with one or more phonemes, the probability module 426 labels the extracted visual speech vectors with one or more previously stored phonemes. Again, each phoneme associated with one or more visual speech feature vectors has a probability associated therewith indicating the likelihood that it was that particular acoustic unit that was spoken in the video segment being considered. Thus, the probability module yields likelihood scores for each considered phoneme in the form of the probability that, given a particular phoneme or acoustic unit (au), the acoustic unit represents the uttered speech characterized by one or more visual speech feature vectors V or, in other words, P(V|acoustic unit). Alternatively, the visual speech feature vectors may be labeled with visemes which, as previously mentioned, are visual phonemes or canonical mouth shapes that accompany speech utterances. Maes at 18:64-20:10: Next, the probabilities generated by modules 416 and 426 are jointly used by A, V probability module 430. In module 430, the respective probabilities from modules 416 and 426 are combined based on a confidence measure 432. Confidence estimation refers to a likelihood or other confidence measure being determined with regard to the recognized input. Recently, efforts have been initiated to develop appropriate confidence measures for recognized speech. In LVCSR Hub5 Workshop, Apr. 29-May 1, 1996, MITAGS, MD, organized by NIST and DARPA, different approaches are proposed to attach to each word a confidence level. A first method uses decision trees trained on word-dependent features (amount of training utterances, minimum and average

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	triphone occurrences, occurrence in language model training, number of phonemes/lefemes, duration, acoustic score (fast match and detailed match), speech or non-speech), sentence-dependent features (signal-to-noise ratio, estimates of speaking rates: number of words or of lefemes or of vowels per second, sentence likelihood provided by the language model, trigram occurrence in the language model), word in a context features (trigram occurrence in language model) as well as speaker profile features (accent, dialect, gender, age, speaking rate, identity, audio quality, SNR, etc . . .). A probability of error is computed on the training data for each of the leaves of the tree. Algorithms to build such trees are disclosed, for example, in Breiman et al., “Classification and regression trees,” Chapman & Hall,1993. At recognition, all or some of these features are measured during recognition and for each word the decision tree is walked to a leave which provides a confidence level. In C. Neti, S. Roukos and E. Eide “Word based confidence measures as a guide for stack search in speech recognition,” ICASSP97, Munich, Germany, April, 1997, is described a method relying entirely on scores returned by IBM stack decoder (using log-likelihood—actually the average incremental log-likelihood, detailed match, fast match). In the LVCSR proceeding, another method to estimate the confidence level is done using predictors via linear regression. The predictor used are: the word duration, the language model score, the average acoustic score (best score) per frame and the fraction of the N-Best list with the same word as top choice. The present embodiment preferably offers a combination of these two approaches (confidence level measured via decision trees and via linear predictors) to systematically extract the confidence level in any translation process, not limited to speech recognition. Another method to detect incorrectly recognized words is disclosed in U.S. Pat. No. 5,937,383 entitled “Apparatus and Methods for Speech Recognition Including Individual or Speaker Class Dependent Decoding History Caches for Fast Word Acceptance or Rejection,” the disclosure of which is incorporated herein by reference. Thus, based on the confidence measure, the probability module 430 decides which probability, i.e., the probability from the visual information path or the probability from the audio information path, to rely on more. This determination may be represented in the following manner:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	$w_1 v_P + w_2 a_P$. It is to be understood that vP represents a probability associated with the visual information, aP represents a probability associated with the corresponding audio information, and w1 and w2 represent respective weights. Thus, based on the confidence measure 432, the module 430 assigns appropriate weights to the probabilities. For instance, if the surrounding environmental noise level is particularly high, i.e., resulting in a lower acoustic confidence measure, there is more of a chance that the probabilities generated by the acoustic decoding path contain errors. Thus, the module 430 assigns a lower weight for w2 than for w1 placing more reliance on the decoded information from the visual path. However, if the noise level is low and thus the acoustic confidence measure is relatively higher, the module may set w2 higher than w1. Alternatively, a visual confidence measure may be used. It is to be appreciated that the first joint use of the visual information and audio information in module 430 is referred to as decision or score fusion. An alternative embodiment implements feature fusion as described in the above-referenced U.S. patent application identified as Ser. No. 09/369,707. Maes at 20:11-20: Then, a search is performed in search module 434 with language models (LM) based on the weighted probabilities received from module 430. That is, the acoustic units identified as having the highest probabilities of representing what was uttered in the arbitrary content video are put together to form words. The words are output by the search engine 434 as the decoded system output. A conventional search engine may be employed. This output is provided to the dialog manager 18 of FIG. 1 for use in disambiguating the user's intent, as described above. Maes at 20:21-21:30: In a preferred embodiment, the audio-visual speech recognition module of FIG. 4 also includes an event detection module 428. As previously mentioned, one problem of conventional speech recognition systems is there inability to discriminate between extraneous audible activity, e.g.,

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	background noise or background speech not intended to be decoded, and speech that is indeed intended to be decoded. This causes such problems as misfiring of the system and “junk” recognition. According to various embodiments, the module may use information from the video path only, information from the audio path only, or information from both paths simultaneously to decide whether or not to decode information. This is accomplished via the event detection module 428. It is to be understood that “event detection” refers to the determination of whether or not an actual speech event that is intended to be decoded is occurring or is going to occur. Based on the output of the event detection module, microphone 406 or the search engine 434 may be enabled/disabled. Note that if no face is detected, then the audio can be processed to make decisions. Referring now to FIG. 5C, an illustrative event detection method using information from the video path only to make the detection decision is shown. To make this determination, the event detection module 428 receives input from the frontal pose detector 420, the visual feature extractor 424 (via the pose normalization block 426), and the audio feature extractor 414. First, in step 510, any mouth openings on a face identified as “frontal” are detected. This detection is based on the tracking of the facial features associated with a detected frontal face, as described in detail above with respect to modules 418 and 420. If a mouth opening or some mouth motion is detected, microphone 406 is turned on, in step 512. Once the microphone is turned on, any signal received therefrom is stored in a buffer (step 514). Then, mouth opening pattern recognition (e.g., periodicity) is performed on the mouth movements associated with the buffered signal to determine if what was buffered was in fact speech (step 516). This is determined by comparing the visual speech feature vectors to pre-stored visual speech patterns consistent with speech. If the buffered data is tagged as speech, in step 518, the buffered data is sent on through the acoustic path so that the buffered data may be recognized, in step 520, so as to yield a decoded output. The process is repeated for each subsequent portion of buffered data until no more mouth openings are detected. In such case, the microphone is then turned off. It is to be understood that FIG. 5C depicts one example of how visual information (e.g., mouth openings) is used to decide whether or not to decode an input audio signal. The event detection module may alternatively control the search module 434, e.g., turning it on or off, in response to whether or not a speech event is

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	detected. Thus, the event detection module is generally a module that decides whether an input signal captured by the microphone is speech given audio and corresponding video information or, P(Speech|A, V). It is also to be appreciated that the event detection methodology may be performed using the audio path information only. In such case, the event detection module 428 may perform one or more speech-only based detection methods such as, for example: signal energy level detection (e.g., is audio signal above a given level); signal zero crossing detection (e.g., are there high enough zero crossings); voice activity detection (non-stationarity of the spectrum) as described in, e.g., N. R. Garner et al., “Robust noise detection for speech recognition and enhancement,” Electronics letters, February 1997, vol. 33, no. 4, pp. 270-271; D. K. Freeman et al., “The voice activity detector of the pan-European digital mobile telephone service, IEEE 1989, CH2673-2; N. R. Garner, “Speech detection in adverse mobile telephony acoustic environments,” to appear in Speech Communications; B. S Atal et al., “A pattern recognition approach to voiced-unvoiced-silence classification with applications to speech recognition, IEEE Trans. Acoustic, Speech and Signal Processing, vol. ASSP-24 n3, 1976. See also, L. R. Rabiner, “Digital processing of speech signals,” Prentice-hall, 1978. Maes at 21:43-64: Thus, in step 522, the uttered speech to be verified may be decoded by classical speech recognition techniques so that a decoded script and associated time alignments are available. This is accomplished using the feature data from the acoustic feature extractor 414. Contemporaneously, in step 524, the visual speech feature vectors from the visual feature extractor 422 are used to produce a visual phonemes (visemes) sequence. Next, in step 526, the script is aligned with the visemes. A rapid (or other) alignment may be performed in a conventional manner in order to attempt to synchronize the two information streams. For example, in one embodiment, rapid alignment as disclosed in the U.S. patent application identified as Ser. No. 09/015,150 and entitled “Apparatus and Method for Generating Phonetic Transcription from Enrollment Utterances,” the disclosure of which is incorporated by reference herein, may be employed. Then, in step 528, a likelihood on the alignment is computed

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	to determine how well the script aligns to the visual data. The results of the likelihood are then used, in step 530, to decide whether an actual speech event occurred or is occurring and whether the information in the paths needs to be recognized. Maes at 21:65-22:39: The audio-visual speech recognition module of FIG. 4 may apply one of, a combination of two of, or all three of, the approaches described above in the event detection module 428 to perform event detection. Video information only based detection is useful so that the module can do the detection when the background noise is too high for a speech only decision. The audio only approach is useful when speech occurs without a visible face present. The combined approach offered by unsupervised utterance verification improves the decision process when a face is detectable with the right pose to improve the acoustic decision. Besides minimizing or eliminating recognition engine misfiring and/or “junk” recognition, the event detection methodology provides better modeling of background noise, that is, when no speech is detected, silence is detected. Also, for embedded applications, such event detection provides additional advantages. For example, the CPU associated with an embedded device can focus on other tasks instead of having to run in a speech detection mode. Also, a battery power savings is realized since speech recognition engine and associated components may be powered off when no speech is present. Other general applications of this speech detection methodology include: (i) use with visible electromagnetic spectrum image or non-visible electromagnetic spectrum image (e.g., far IR) camera in vehicle-based speech detection or noisy environment; (ii) speaker detection in an audience to focus local or array microphones; (iii) speaker recognition (as in the above-referenced U.S. patent application Ser. No. 09/369,706 and tagging in broadcast news or TeleVideo conferencing. One of ordinary skill in the art will contemplate other applications given the inventive teachings described herein. It is to be appreciated that the audio-visual speech recognition module of FIG. 4 may employ the alternative embodiments of audio-visual speech detection and recognition described in the above-referenced U.S. patent application identified as Ser. No. 09/369,707. For instance, whereas the

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	embodiment of FIG. 4 illustrates a decision or score fusion approach, the module may employ a feature fusion approach and/or a serial rescoring approach, as described in the above-referenced U.S. patent application identified as Ser. No. 09/369,707. Maes at 22:41-67: Referring now to FIG. 6, a block diagram illustrates a preferred embodiment of an audio-visual speaker recognition module that may be employed as one of the recognition modules of FIG. 1 to perform speaker recognition using multi-modal input data received in accordance with the invention. It is to be appreciated that such an audio-visual speaker recognition module is disclosed in the above-referenced U.S. patent application identified as Ser. No. 09/369,706, filed on Aug. 6, 1999 and entitled “Methods And Apparatus for Audio-Visual Speaker Recognition and Utterance Verification.” A description of one of the embodiments of such an audio-visual speaker recognition module for use in a preferred embodiment of the multi-modal conversational computing system of the invention is provided below in this section. However, it is to be appreciated that other mechanisms for performing speaker recognition may be employed. The audio-visual speaker recognition and utterance verification module shown in FIG. 6 uses a decision fusion approach. Like the audio-visual speech recognition module of FIG. 4, the speaker recognition module of FIG. 6 may receive the same types of arbitrary content video from the camera 604 and audio from the microphone 606 via the I/O manager 14. While the camera and microphone have different reference numerals in FIG. 6 than in FIG. 4, it is to be appreciated that they may be the same camera and microphone. Maes at 23:10-23: The module includes an auditory feature extractor 614. The feature extractor 614 receives an audio or speech signal and, as is known in the art, extracts spectral features from the signal at regular intervals. The spectral features are in the form of acoustic feature vectors (signals) which are then passed on to an audio speaker recognition module 616. Before acoustic vectors are extracted, the speech signal may be sampled at a rate of 16 kilohertz (kHz). A frame may consist of a segment of

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	speech having a 25 millisecond (msec) duration. In such an arrangement, the extraction process preferably produces 24 dimensional acoustic cepstral vectors via the process described below. Frames are advanced every 10 msec to obtain succeeding acoustic vectors. Of course, other front-ends may be employed. Maes at 23:44-24:46: After the acoustic feature vectors, denoted in FIG. 6. by the letter A, are extracted, they are provided to the audio speaker recognition module 616. It is to be understood that the module 616 may perform speaker identification and/or speaker verification using the extracted acoustic feature vectors. The processes of speaker identification and verification may be accomplished via any conventional acoustic information speaker recognition system. For example, speaker recognition module 616 may implement the recognition techniques described in the U.S. patent application identified by Ser. No. 08/788,471, filed on Jan. 28, 1997, and entitled: “Text Independent Speaker Recognition for Transparent Command Ambiguity Resolution and Continuous Access Control,” the disclosure of which is incorporated herein by reference. An illustrative speaker identification process for use in module **616** will now be described. The illustrative system is disclosed in H. Beigi, S. H. Maes, U. V. Chaudari and J. S. Sorenson, “IBM model-based and frame-by-frame speaker recognition,” Speaker Recognition and its Commercial and Forensic Applications, Avignon, France 1998. The illustrative speaker identification system may use two techniques: a model-based approach and a frame-based approach. In the experiments described herein, we use the frame-based approach for speaker identification based on audio. The frame-based approach can be described in the following manner. Let M_i be the model corresponding to the ith enrolled speaker. M_i is represented by a mixture Gaussian model defined by the parameter set $\{\mu_{i,j}, \Sigma_{i,j}, p_{i,j}\}_{j=1,\ldots n_i}$, consisting of the mean vector, covariance matrix and mixture weights for each of the n_i components of speaker i's model. These models are created using training data consisting of a sequence of K frames of speech with d-dimensional cepstral feature vectors, $\{f_m\}m=1, \ldots K$. The goal of speaker identification is to find the model, M_i, that best explains the test data represented by a sequence of N frames, $\{f_n\}n=1, \ldots$

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	N. We use the following frame-based weighted likelihood distance measure, $d_{i,n}$, in making the decision: $$d_{i,n} = -\log\left[\sum_{j=1}^{n_i} p_{i,j}\, p(f_n \mid \mu_{i,j}, \Sigma_{i,j})\right].$$ The total distance D_i of model M_i from the test data is then taken to be the sum of the distances over all the test frames: $$D_i = \sum_{n=1}^{N} d_{i,n}.$$ Thus, the above approach finds the closest matching model and the person whose model that represents is determined to be the person whose utterance is being processed. Speaker verification may be performed in a similar manner, however, the input acoustic data is compared to determine if the data matches closely enough with stored models. If the comparison yields a close enough match, the person uttering the speech is verified. The match is accepted or rejected by comparing the match with competing models. These models can be selected to be similar to the claimant speaker or be speaker independent (i.e., a single or a set of speaker independent models). If the claimant wins and wins with enough margin (computed at the level of the likelihood or the distance to the models), we accept the claimant. Otherwise, the claimant is rejected. It should be understood that, at enrollment, the input speech is collected for a speaker to build the mixture gaussian model M_i that characterize each speaker. Maes at 24:46-25:3:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Referring now to the video signal path II of FIG. 6, the methodologies of processing visual information will now be explained. The audio-visual speaker recognition and utterance verification module includes an active speaker face segmentation module 620 and a face recognition module 624. The active speaker face segmentation module 620 receives video input from camera 604. It is to be appreciated that speaker face detection can also be performed directly in the compressed data domain and/or from audio and video information rather than just from video information. In any case, segmentation module 620 generally locates and tracks the speaker's face and facial features within the arbitrary video background. This will be explained in detail below. From data provided from the segmentation module 622, an identification and/or verification operation may be performed by recognition module 624 to identify and/or verify the face of the person assumed to be the speaker in the video. Verification can also be performed by adding score thresholding or competing models. Thus, the visual mode of speaker identification is implemented as a face recognition system where faces are found and tracked in the video sequences, and recognized by comparison with a database of candidate face templates. As will be explained later, utterance verification provides a technique to verify that the person actually uttered the speech used to recognize him. Maes at 26:39-53: Next, the results of the face recognition module 624 and the audio speaker recognition module 616 are provided to respective confidence estimation blocks 626 and 618 where confidence estimation is performed. Confidence estimation refers to a likelihood or other confidence measure being determined with regard to the recognized input. In one embodiment, the confidence estimation procedure may include measurement of noise levels respectively associated with the audio signal and the video signal. These levels may be measured internally or externally with respect to the system. A higher level of noise associated with a signal generally means that the confidence attributed to the recognition results associated with that signal is lower. Therefore, these confidence measures are taken into consideration during the weighting of the visual and acoustic results discussed below. Maes at 28:11-35:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Thus, the joint identification/verification module 630 makes a decision with regard to the speaker. In a verification scenario, based on one of the techniques described above, a decision may be made to accept the speaker if he is verified via both the acoustic path and the visual path. However, he may be rejected if he is only verified through one of the paths. In an identification scenario, for example, the top three scores from the face identification process may be combined with the top three scores from the acoustic speaker identification process. Then, the highest combined score is identified as the speaker. In a preferred embodiment, before the module makes a final disposition with respect to the speaker, the system performs an utterance verification operation. It is to be appreciated that utterance verification is performed by the utterance verification module 628 (FIG. 6) based on input from the acoustic feature extractor 614 and a visual speech feature extractor 622. Before describing utterance verification, a description of illustrative techniques for extracting visual speech feature vectors will follow. Particularly, the visual speech feature extractor 622 extracts visual speech feature vectors (e.g., mouth or lip-related parameters), denoted in FIG. 6 as the letter V, from the face detected in the video frame by the active speaker face segmentation module 620. Maes at 29:22-30:39: Given the extracted visual speech feature vectors (V) from extractor 622 and the acoustic feature vectors(A) from extractor 614, the AV utterance verifier 628 performs verification. Verification may involve a comparison of the resulting likelihood, for example, of aligning the audio on a random sequence of visemes. As is known, visemes, or visual phonemes, are generally canonical mouth shapes that accompany speech utterances which are categorized and pre-stored similar to acoustic phonemes. A goal associated with utterance verification is to make a determination that the speech used to verify the speaker in the audio path I and the visual cues used to verify the speaker in the video path II correlate or align. This allows the system to be confident that the speech data that is being used to recognize the speaker is actually what the speaker uttered. Such a determination has many advantages. For example, from the utterance verification, it can be determined whether the user is lip synching to a pre-recorded tape playback to attempt to fool the

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	system. Also, from utterance verification, errors in the audio decoding path may be detected. Depending on the number of errors, a confidence measure may be produced and used by the system. Referring now to FIG. 7, a flow diagram of an utterance verification methodology is shown. Utterance verification may be performed in: (i) a supervised mode, i.e., when the text (script) is known and available to the system; or (ii) an unsupervised mode, i.e., when the text (script) is not known and available to the system. Thus, in step 702A (unsupervised mode), the uttered speech to be verified may be decoded by classical speech recognition techniques so that a decoded script and associated time alignments are available. This is accomplished using the feature data from the acoustic feature extractor 614. Contemporaneously, in step 704, the visual speech feature vectors from the visual feature extractor 622 are used to produce a visual phonemes or visemes sequence. Next, in step 706, the script is aligned with the visemes. A rapid (or other) alignment may be performed in a conventional manner in order to attempt to synchronize the two information streams. For example, in one embodiment, rapid alignment as disclosed in the U.S. patent application identified by Ser. No. 09/015,150 and entitled “Apparatus and Method for Generating Phonetic Transcription from Enrollment Utterances,” the disclosure of which is incorporated by reference herein, may be employed. Note that in a supervised mode, step 702B replaces step 702A such that the expected or known script is aligned with the visemes in step 706, rather than the decoded version of the script. Then, in step 708, a likelihood on the alignment is computed to determine how well the script aligns to the visual data. The results of the likelihood are then provided to a decision block 632 which, along with the results of the score module 630, decides on a final disposition of the speaker, e.g., accept him or reject him. This may be used to allow or deny access to a variety of devices, applications, facilities, etc. So, in the unsupervised utterance verification mode, the system is able to check that the user is indeed speaking rather than using a playback device and moving his lips. Also, a priori, errors may be detected in the audio decoding. In the supervised mode, the system is able to prove that the user

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	uttered the text if the recognized text is sufficiently aligned or correlated to the extracted lip parameters. It is to be appreciated that utterance verification in the unsupervised mode can be used to perform speech detection as disclosed in the above-referenced U.S. patent application identified as U.S. Ser. No. 09/369,707. Indeed, if acoustic and visual activities are detected, they can be verified against each other. When the resulting acoustic utterance is accepted, the system considers that speech is detected. Otherwise, it is considered that extraneous activities are present. It is to be appreciated that the audio-visual speaker recognition module of FIG. 6 may employ the alternative embodiments of audio-visual speaker recognition described in the above-referenced U.S. patent application identified as Ser. No. 09/369,706. For instance, whereas the embodiment of FIG. 6 illustrates a decision or score fusion approach, the module 20 may employ a feature fusion approach and/or a serial rescoring approach, as described in the above-referenced U.S. patent application identified as Ser. No. 09/369,706. It is to be further appreciated that the output of the audio-visual speaker recognition system of FIG. 6 is provided to the dialog manager 18 of FIG. 1 for use in disambiguating the user's intent, as explained above. Maes at 30:41-31:34: Referring now to FIGS. 8A and 8B, block diagrams illustrate a preferred embodiment of a conversational virtual machine (CVM). It is to be appreciated that such a conversational virtual machine is disclosed in the above-referenced PCT international patent application identified as US99/22927 filed on Oct. 1, 1999 and entitled “Conversational Computing Via Conversational Virtual Machine.” A description of one of the embodiments of such a machine for use in a preferred embodiment of the multi-modal conversational computing system of the present invention is provided below in this section. However, it is to be appreciated that other mechanisms for implementing conversational computing according to the invention may be employed, as explained below.

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	It is to be understood that the CVM described below may be employed to provide a framework for: portions of the I/O subsystem 12; I/O manager 14; recognition engines 16; dialog manager 18; and context stack 20 of FIG. 1. Throughout the description of the CVM below, the components of the CVM that may be employed to implement these functional components of FIG. 1 will be noted. However, while the CVM may be used because of its ability to implement an I/O manager, a modality independent context manager (context stack), a dialog manager (when disambiguation is performed), a classifier (when mood or focus is determined), required engines and APIs/interfaces to the dialog manager to run applications, it is important to note that other mechanisms may be alternatively used to implement these functional components of a multi-modal conversational computing system of the invention. For example, functional components of a multi-modal conversational computing system of the invention may be implemented through a browser that carries these functions, an OSS (operating system service) layer, a VM (virtual machine) or even just an application that implements all these functionalities, possibly without explicitly identifying these component but rather by implementing hard-coded equivalent services. It is also to be appreciated that the implementation may support only modalities of speech and video and, in such a case, does not need to support other modalities (e.g., handwriting, GUI, etc.). Thus, the CVM may be employed as a main component for implementing conversational computing according to the conversational computing paradigm described above with respect to the present invention. In one embodiment, the CVM is a conversational platform or kernel running on top of a conventional OS (operating system) or RTOS (real-time operating system). A CVM platform can also be implemented with PvC (pervasive computing) clients as well as servers and can be distributed across multiple systems (clients and servers). In general, the CVM provides conversational APIs (application programming interfaces) and protocols between conversational subsystems (e.g., speech recognition engine, text-to speech, etc.) and conversational and/or conventional applications. The CVM may also provide backward compatibility to existing applications, with a more limited interface. As discussed in detail below, the CVM provides conversational services and behaviors as well as conversational protocols for interaction with multiple applications and devices also equipped with a CVM layer, or at least, conversationally aware.

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Maes at 31:43-32:45: A CVM (or operating system) based on the conversational computing paradigm described herein allows a computer or any other interactive device to converse with a user. The CVM further allows the user to run multiple tasks on a machine regardless if the machine has no display or GUI capabilities, nor any keyboard, pen or pointing device. Indeed, the user can manage these tasks like a conversation and bring a task or multiple simultaneous tasks, to closure. To manage tasks like a conversation, the CVM affords the capability of relying on mixed initiatives, contexts and advanced levels of abstraction, to perform its various functions. Mixed initiative or free flow navigation allows a user to naturally complete, modify, or correct a request via dialog with the system. Mixed initiative also implies that the CVM can actively help (take the initiative to help) and coach a user through a task, especially in speech-enabled applications, wherein the mixed initiative capability is a natural way of compensating for a display-less system or system with limited display capabilities. In general, the CVM complements conventional interfaces and user input/output rather than replacing them. This is the notion of “multi-modality” whereby speech, and video as described above, may be used in parallel with a mouse, keyboard, and other input devices such as a pen. Conventional interfaces can be replaced when device limitations constrain the implementation of certain interfaces. In addition, the ubiquity and uniformity of the resulting interface across devices, tiers and services is an additional mandatory characteristic. It is to be understood that a CVM system can, to a large extent, function with conventional input and/or output media. Indeed, a computer with classical keyboard inputs and pointing devices coupled with a traditional monitor display can profit significantly by utilizing the CVM. One example is described in U.S. patent application identified as U.S. Ser. No. 09/507,526 filed on Feb. 18, 2000 and entitled “Multi-Modal Shell” which claims priority to U.S. provisional patent application identified as U.S. Ser. No. 60/128,081 filed on Apr. 7, 1999 and U.S. provisional patent application identified by Ser. No. 60/158,777 filed on Oct. 12, 1999, the disclosures of which are incorporated by reference herein (which describes a method for constructing a true multi-modal application with tight synchronization between a GUI modality and a speech modality). In other words, even users who do not want to talk to their computer can also realize a dramatic positive change to their interaction with the CVM enabled machine.

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Referring now to FIG. 8A, a block diagram illustrates a CVM system according to a preferred embodiment, which may be implemented on a client device or a server. In terms of the vehicle example above, this means that the components of the system 10 may be located locally (in the vehicle), remotely (e.g., connected wirelessly to the vehicle), or some combination thereof. In general, the CVM provides a universal coordinated multi-modal conversational user interface (CUI) 780. The “multi-modality” aspect of the CUI implies that various I/O resources such as voice, keyboard, pen, and pointing device (mouse), keypads, touch screens, etc., and video as described above, can be used in conjunction with the CVM platform. The “universality” aspect of the CUI implies that the CVM system provides the same UI (user interface) to a user whether the CVM is implemented in connection with a desktop computer, a PDA with limited display capabilities, or with a phone where no display is provided. In other words, universality implies that the CVM system can appropriately handle the UI of devices with capabilities ranging from speech only to multi-modal, i.e., speech + GUI, to purely GUI. As per the present invention, the system may be extended to include video input data as well. Therefore, the universal CUI provides the same UI for all user interactions, regardless of the access modality. Maes at 32:60-33:25: The CVM system can run a plurality of applications including conversationally aware applications 782 (i.e., applications that “speak” conversational protocols) and conventional applications 784. The conversationally aware applications 782 are applications that are specifically programmed for operating with a CVM core layer (or kernel) 788 via conversational application APIs 786. In general, the CVM kernel 788 controls the dialog across applications and devices on the basis of their registered conversational capabilities and requirements and provides a unified conversational user interface which goes far beyond adding speech as I/O modality to provide conversational system behaviors. The CVM system may be built on top of a conventional OS and APIs 790 and conventional device hardware 792 and located on a server or any client device (PC, PDA, PvC). The conventional applications 784 are managed by the CVM kernel layer 788 which is responsible for accessing, via the OS APIs, GUI menus and commands of the conventional applications as well as the underlying OS commands. The CVM automatically handles all the input/output issues, including the conversational subsystems 796 (i.e., conversational engines) and conventional

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	subsystems (e.g., file system and conventional drivers) of the conventional OS 790. In general, conversational sub-systems 796 are responsible for converting voice requests into queries and converting outputs and results into spoken messages using the appropriate data files 794 (e.g., contexts, finite state grammars, vocabularies, language models, symbolic query maps, etc.). The conversational application API 786 conveys all the information for the CVM 788 to transform queries into application calls and conversely converts output into speech, appropriately sorted before being provided to the user. Maes at 33:26-34:11: Referring now to FIG. 8B, a diagram illustrates abstract programming layers of a CVM according to a preferred embodiment. The abstract layers of the CVM comprise conversationally aware applications 800 and conventional applications 801 that can run on top of the CVM. An application that relies on multi-modal disambiguation is an example of such a conversational application that executes on top of the CVM. Similarly, an application that exploits focus information or mood can be considered as a conversational application on top of the CVM. These applications are the programs that are executed by the system to provide the user with the interaction he desires within the environment in which the system is deployed. As discussed above, the conversationally aware applications 800 interact with a CVM kernel layer 802 via a conversational application API layer 803. The conversational application API layer 803 encompasses conversational programming languages/scripts and libraries (conversational foundation classes) to provide the various features offered by the CVM kernel 802. For example, the conversational programming languages/scripts provide the conversational APIs that allow an application developer to hook (or develop) conversationally aware applications 800. They also provide the conversational API layer 803, conversational protocols 804 and system calls that allow a developer to build the conversational features into an application to make it “conversationally aware.” The code implementing the applications, API calls and protocol calls includes interpreted and compiled scripts and programs, with library links, conversational logic engine call and conversational foundation classes.

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	More specifically, the conversational application API layer 803 comprises a plurality of conversational foundation classes 805 (or fundamental dialog components) which are provided to the application developer through library functions that may be used to build a CUI or conversationally aware applications 800. The conversational foundation classes 805 are the elementary components or conversational gestures (as described by T. V. Raman, in “Auditory User Interfaces, Toward The Speaking Computer,” Kluwer Academic Publishers, Boston 1997) that characterize any dialog, independently of the modality or combination of modalities (which can be implemented procedurally or declaratively). The conversational foundation classes 805 comprise CUI building blocks and conversational platform libraries, dialog modules and components, and dialog scripts and beans. The conversational foundation classes 805 may be compiled locally into conversational objects 806. More specifically, the conversational objects 805 (or dialog components) are compiled from the conversational foundation classes 805 (fundamental dialog components) by combining the different individual classes in a code calling these libraries through a programming language such as Java or C++. Maes at 34:34-58: Some examples of conversational gestures that may be implemented are as follows. A conversational gesture message is used by a machine to convey informational messages to the user. The gesture messages will typically be rendered as a displayed string or spoken prompt. Portions of the message to be spoken can be a function of the current state of the various applications/dialogs running on top of the CVM. A conversational gesture “select from set” is used to encapsulate dialogues where the user is expected to pick from a set of discrete choices. It encapsulates the prompt, the default selection, as well as the set of legal choices. Conversational gesture message “select from range” encapsulates dialogs where the user is allowed to pick a value from a continuous range of values. The gesture encapsulates the valid range, the current selection, and an informational prompt. In addition, conversational gesture input is used to obtain user input when the input constraints are more complex (or perhaps non-existent). The gesture encapsulates the user prompt, application-level semantics about the item of information being requested and possibly a predicate to test the validity of the input. As described above, however, the conversational foundation classes include, yet surpass, the concept of conversational gestures (i.e.,

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	they extend to the level of fundamental behavior and services as well as rules to perform conversational tasks). Maes at 35:3-53: The development environment offered by the CVM is referred to herein as SPOKEN AGE™. Spoken Age allows a developer to build, simulate and debug conversationally aware applications for CVM. Besides offering direct implementation of the API calls, it offers also tools to build advanced conversational interfaces with multiple personalities, voice fonts which allow the user to select the type of voice providing the output, and conversational formatting languages which build conversational presentations like Postcript and AFL (audio formatting languages). As described above, the conversational application API layer 803 encompasses conversational programming languages and scripts to provide universal conversational input and output, conversational logic and conversational meta-information exchange protocols. The conversational programming language/scripts allow to use any available resources as input or output stream. Using the conversational engines 808 (recognition engines 16 of FIG. 1) and conversational data files 809 (accessed by CVM 802 via conversation engine APIs 807), each input is converted into a binary or ASCII input, which can be directly processed by the programming language as built-in objects. Calls, flags and tags can be automatically included to transmit between object and processes the conversational meta-information required to correctly interface with the different objects. Moreover, output streams can be specially formatted according to the needs of the application or user. These programming tools allow multi-modal discourse processing to be readily built. Moreover, logic statement status and operators are expanded to handle the richness of conversational queries that can be compared on the bases of their ASCII/binary content or on the basis of their NLU-converted (natural language understanding-converted) query (input/output of conventional and conversational sub-systems) or FSG-based queries (where the system used restricted commands). Logic operators can be implemented to test or modify such systems. Conversational logic values/operators expand to include: true, false, incomplete, ambiguous, different/equivalent for an ASCII point of view, different/equivalent from a NLU point of view,

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	different/equivalent from a active query field point of view, unknown, incompatible, and incomparable. Furthermore, the conversational application API layer 803 comprises code for providing extensions of the underlying OS features and behavior. Such extensions include, for example, high level of abstraction and abstract categories associated with any object, self-registration mechanisms of abstract categories, memorization, summarization, conversational search, selection, redirection, user customization, train ability, help, multi-user and security capabilities, as well as the foundation class libraries. Maes at 35:54-36:7: The conversational computing system of FIG. 8B further comprises a conversational engine API layer 807 which provides an interface between core engines conversational engines 808 (e.g., speech recognition, speaker recognition, NL parsing, NLU, TTS and speech compression/decompression engines, visual recognition) and the applications using them. The engine API layer 807 also provides the protocols to communicate with core engines whether they be local or remote. An I/O API layer 810 provides an interface with conventional I/O resources 811 such as a keyboard, mouse, touch screen, keypad, etc. (for providing a multi-modal conversational UI), an audio subsystem for capturing speech I/O (audio in/audio out), and a video subsystem for capturing video I/O. The I/O API layer 810 provides device abstractions, I/O abstractions and UI abstractions. The I/O resources 811 will register with the CVM kernel layer 802 via the I/O API layer 810. It is to be understood that the I/O APIs 810 may be implemented as part of the I/O manager 14 of FIG. 1, while the I/O resources 811 may be implemented as part of the I/O subsystem 12 of FIG. 1. Maes at 36:8-48: The core CVM kernel layer 802 comprises programming layers such as a conversational application and behavior/service manager layer 815, a conversational dialog manager (arbitrator) layer 819, a conversational resource manager layer 820, a task/dispatcher manager 821 and a

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	meta-information manager 822, which provide the core functions of the CVM layer 802. It is to be understood that these components may be implemented as part of the dialog manager 18 of FIG. 1. The conversational application and behavior/service manager layer 815 comprises functions for managing the conventional and conversationally aware applications 800 and 801. Such management functions include, for example, keeping track of which applications are registered (both local and network-distributed), what are the dialog interfaces (if any) of the applications, and what is the state of each application. In addition, the conversational application and services/behavior manager 815 initiates all the tasks associated with any specific service or behavior provided by the CVM system. The conversational services and behaviors are all the behaviors and features of a conversational UI that the user may expect to find in the applications and interactions, as well as the features that an application developer may expect to be able to access via APIs (without having to implement with the development of the application). Examples of the conversational services and behavior provided by the CVM kernel 802 include, but are not limited to, conversational categorization and meta-information, conversational object, resource and file management, conversational search, conversational selection, conversational customization, conversational security, conversational help, conversational prioritization, conversational resource management, output formatting and presentation, summarization, conversational delayed actions/agents/memorization, conversational logic, and coordinated interfaces and devices. Such services are provided through API calls via the conversational application API Layer 803. The conversational application and behavior/services manager 815 is responsible for executing all the different functions needed to adapt the UI to the capabilities and constraints of the device, application and/or user preferences. Maes at 36:49-37:3: The conversational dialog manager 819 comprises functions for managing the dialog (conversational dialog comprising speech and other multi-modal I/O such as GUI keyboard, pointer, mouse, as well as video input, etc.) and arbitration (dialog manager arbitrator or DMA) across all registered applications. In particular, the conversational dialog manager 819 determines what information the user has, which inputs the user presents, and which application(s) should handle the user inputs. The DMA processes abstracted I/O events (abstracted by the I/O manager)

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	using the context/history to understand the user intent. When an abstract event occurs, the DMA determines the target of the event and, if needed, seeks confirmation, disambiguation, correction, more details, etc., until the intent is unambiguous and fully determined. The DMA then launches the action associated to the user's query. The DMA function handles multi-modal I/O events to: (1) determine the target application or dialog (or portion of it); and (2) use past history and context to: (a) understand the intent of the user; (b) follow up with a dialog to disambiguate, complete, correct or confirm the understanding; (c) or, dispatch a task resulting from fall understanding of the intent of the user. Maes at 37:17-50: The task dispatcher/manager 821 dispatches and coordinates different tasks and processes that are spawned (by the user and machine) on local and networked conventional and conversational resources. The meta-information manager 822 manages the meta-information associated with the system via a meta-information repository 818. The meta-information manager 822 and repository 818 collect all the information typically assumed known in a conversational interaction but not available at the level of the current conversation. Examples are a-priori knowledge, cultural, educational assumptions and persistent information, past request, references, information about the user, the application, news, etc. It is typically the information that needs to be preserved and persist beyond the length/life of the conversational history/context and the information that is expected to be common knowledge for the conversation and, therefore, has never been defined during the current and possible past conversational interactions. Also, shortcuts to commands, resources and macros, etc., are managed by the meta-information manager 822 and stored in the meta-information repository 818. In addition, the meta-information repository 818 includes a user-usage log based on user identity. It is to be appreciated that services such as conversational help and assistance, as well as some dialog prompts (introduction, questions, feedback, etc.) provided by the CVM system can be tailored based on the usage history of the user as stored in the meta-information repository 818 and associated with the application. If a user has been previously interacting with a given application, an explanation can be reduced assuming that it is familiar to the user. Similarly, if a user commits many errors, the explanations can be more complex, as

Claim 7 of the '957 Patent	Exemplary Citations to Maes
	multiple errors are interpreted as user uncertainty, unfamiliarity, or incomprehension/misunderstanding of the application or function. Maes at 37:51-38:2: A context stack 817 is managed by the dialog manager 819, possibly through a context manager that interacts with the dialog manager and arbitrator. It is to be understood that the context stack 817 may be implemented as part of the context stack 20 of FIG. 1. The context stack 817 comprises all the information associated with an application. Such information includes all the variable, states, input, output and queries to the backend that are performed in the context of the dialog and any extraneous event that occurs during the dialog. The context stack is associated with the organized/sorted context corresponding to each active dialog (or deferred dialog-agents/memorization). A global history 816 is included in the CVM system and includes information that is stored beyond the context of each application. The global history stores, for example, the information that is associated with all the applications and actions taking during a conversational session (i.e., the history of the dialog between user and machine for a current session or from when the machine was activated). Maes at 38:3-17: The CVM kernel layer 802 further comprises a backend abstraction layer 823 which allows access to backend business logic 813 via the dialog manager 819 (rather than bypassing the dialog manager 819). This allows such accesses to be added to the context stack 817 and global history 816. For instance, the backend abstraction layer 823 can translate input and output to and from the dialog manager 819 to database queries. This layer 823 will convert standardized attribute value n-tuples into database queries and translate the result of such queries into tables or sets of attribute value n-tuples back to the dialog manager 819. In addition, a conversational transcoding layer 824 is provided to adapt the behavior, UI and dialog presented to the user based on the I/O and engine capabilities of the device which executes the CVM system. Maes at 38:18-37:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	The CVM system further comprises a communication stack 814 (or communication engines) as part of the underlying system services provided by the OS 812. The CVM system utilizes the communication stack to transmit information via conversational protocols 804 which extend the conventional communication services to provide conversational communication. It is to be understood that the communication stack 814 may be implemented in connection with the well-known OSI (open system interconnection) protocol layers for providing conversational communication exchange between conversational devices. As is known in the art, OSI comprises seven layers with each layer performing a respective function to provide communication between network distributed conversational applications of network-connected devices. Such layers (whose functions are well-understood) comprise an application layer, a presentation layer, a session layer, a transport layer, a network layer, a data link layer and a physical layer. The application layer is extended to allow conversational communication via the conversational protocols 804. Maes at 39:14-43: FIGS. 9A and 9B will be used to generally describe mood/focus classification techniques that may be employed in the mood/focus classifier 22 (FIG. 1) with respect to speech-based event data. However, the extended application to include the modality associated with video-based event data will be illustrated in the context of FIG. 9C where it is shown that these classification techniques can be easily applied to multi-modal input. FIG. 9A depicts an apparatus for collecting data associated with a voice of a user, in accordance with the present invention. The apparatus is designated generally as 900. The apparatus includes a dialog management unit 902 which conducts a conversation with the user. It is to be understood that the user-provided input data events are preferably provided to the system 900 via the I/O manager 14 of FIG. 1. Apparatus 900 further includes an audio capture module 906 which is coupled to the dialog management unit 902 and which captures a speech waveform associated with utterances spoken by the user 904 during the conversation. While shown for ease of explanation in FIG. 9A, the audio capture unit 906 may be part of the I/O subsystem 12 of FIG. 1. In which case, the captured input data is passed onto system 900 via the I/O manager 14. As used

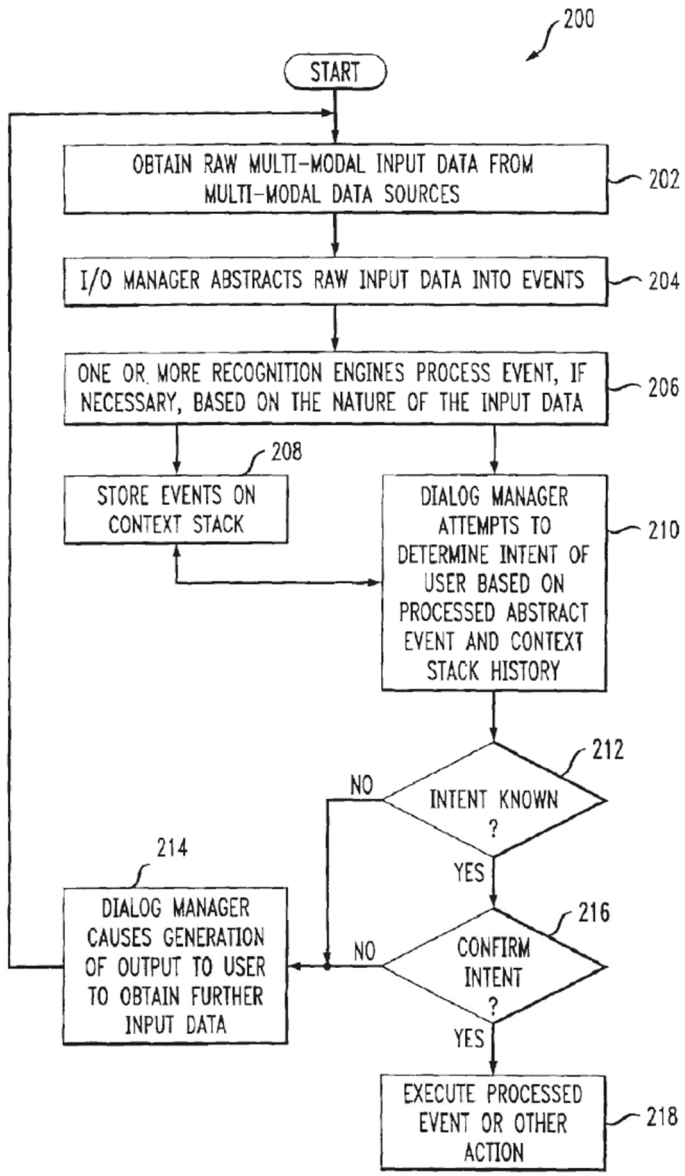
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	herein, a conversation should be broadly understood to include any interaction, between a first human and either a second human, a machine, or a combination thereof, which includes at least some speech. Again, based on the above described teachings of the multi-modal system 10 of the invention, the mood classification (focus detection) system 900 may be extended to process video in a similar manner. Maes at 39:44-40:10: Apparatus 900 further includes an acoustic front end 908 which is coupled to the audio capture module 906 and which is configured to receive and digitize the speech waveform so as to provide a digitized speech waveform. Further, acoustic front end 908 is also configured to extract, from the digitized speech waveform, at least one acoustic feature which is correlated with at least one user attribute. The at least one user attribute can include at least one of the following: gender of the user, age of the user, accent of the user, native language of the user, dialect of the user, socioeconomic classification of the user, educational level of the user, and emotional state of the user. The dialog management unit 902 may employ acoustic features, such as MEL cepstra, obtained from acoustic front end 908 and may therefore, if desired, have a direct coupling thereto. Apparatus 900 further includes a processing module 910 which is coupled to the acoustic front end 908 and which analyzes the at least one acoustic feature to determine the at least one user attribute. Yet further, apparatus 900 includes a data warehouse 912 which is coupled to the processing module 910 and which stores the at least one user attribute, together with at least one identifying indicia, in a form for subsequent data mining thereon. Identifying indicia will be discussed elsewhere herein. The gender of the user can be determined by classifying the pitch of the user's voice, or by simply clustering the features. In the latter method, voice prints associated with a large set of speakers of a given gender are built and a speaker classification is then performed with the two sets of models. Age of the user can also be determined via classification of age groups, in a manner similar to gender. Although having limited reliability, broad classes of ages, such as children, teenagers, adults and senior citizens can be separated in this fashion.

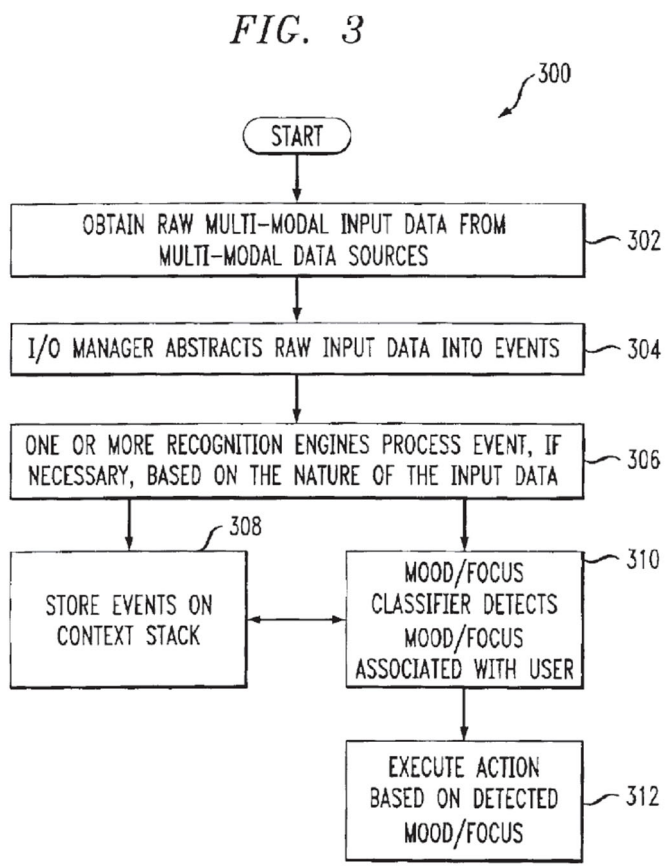
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Maes at 41:48-42:7: Apparatus 900 can further include a post processor 938 which is coupled to the data warehouse 912 and which is configured to transcribe user utterances and to perform keyword spotting thereon. Although shown as a separate item in FIG. 9A, the post processor can be a part of the processing module 910 or of any of the sub-components thereof. For example, it can be implemented as part of the speech recognizor 926. Post processor 938 can be implemented as part of the processor of an IVR, as an application specific circuit, or on a general purpose computer with suitable software modules. Post processor 938 can employ speech recognizor 926. Post processor 938 can also include a semantic module (not shown) to interpret meaning of phrases. The semantic module could be used by speech recognizor 926 to indicate that some decoding candidates in a list are meaningless and should be discarded/replaced with meaningful candidates. The acoustic front end 908 can typically be an eight dimensions plus energy front end as known in the art. However, it should be understood that 13, 24, or any other number of dimensions could be used. MEL cepstra can be computed, for example, over 25 ms frames with a 10 ms overlap, along with the delta and delta delta parameters, that is, the first and second finite derivatives. Such acoustic features can be supplied to the speaker clusterer and classifier 920, speech recognizor 926 and accent identifier 932, as shown in FIG. 9A. Maes at 42:20-23: User queries can be transcribed by an IVR or otherwise. Speech features can first be processed by a text-independent speaker classification system, for example, in speaker clusterer and classifier 920. Maes at 42:57-43:2: The speech recognizor 926 can transcribe the queries from the user. It can be a speaker-independent or class-dependent large vocabulary continuous speech recognition, or system could

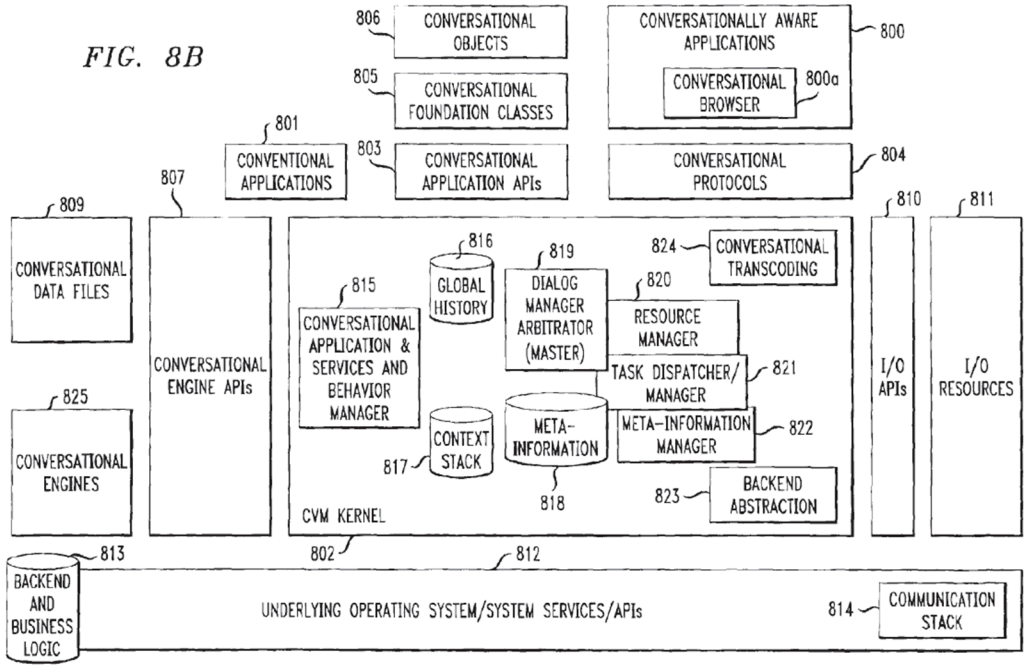
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	be something as simple as a keyword spotter to detect insults (for example) and the like. Such systems are well known in the art. The output can be full sentences, but finer granularity can also be attained; for example, time alignment of the recognized words. The time stamped transcriptions can also be considered as part of the processed features, and will be discussed further below with respect to methods in accordance with the present invention. Thus, conversation from every stage of a transaction can be transcribed and stored. Maes at 45:17-60: Referring now to FIG. 10, a block diagram of an illustrative hardware implementation of a multi-modal conversational computing system according to the invention is shown. In this particular implementation, a processor 1092 for controlling and performing the various operations associated with the illustrative systems of the invention depicted in FIGS. 1 through 9C is coupled to a memory 1094 and a user interface 1096. It is to be appreciated that the term “processor” as used herein is intended to include any processing device, such as, for example, one that includes a CPU (central processing unit) and/or other processing circuitry. For example, the processor may be a digital signal processor, as is known in the art. Also the term “processor” may refer to more than one individual processor. The term “memory” as used herein is intended to include memory associated with a processor or CPU, such as, for example, RAM, ROM, a fixed memory device (e.g., hard drive), a removable memory device (e.g., diskette), a flash memory, etc. In addition, the term “user interface” as used herein is intended to include, for example, one or more input devices, e.g., keyboard, for inputting data to the processing unit, and/or one or more output devices, e.g., CRT display and/or printer, for providing results associated with the processing unit. The user interface 1096 is also intended to include the one or more microphones for receiving user speech and the one or more cameras/sensors for capturing image data, as well as any other I/O interface devices used in the multi-modal system. To the extent it is found that Maes does not expressly disclose this limitation, the limitation is inherent. And to the extent it is found that Maes does not anticipate this claim, Maes renders it obvious, standing alone, in combination with the knowledge of a person of ordinary skill in the art, and/or in combination with one or more other prior art references identified in the Cover Pleading

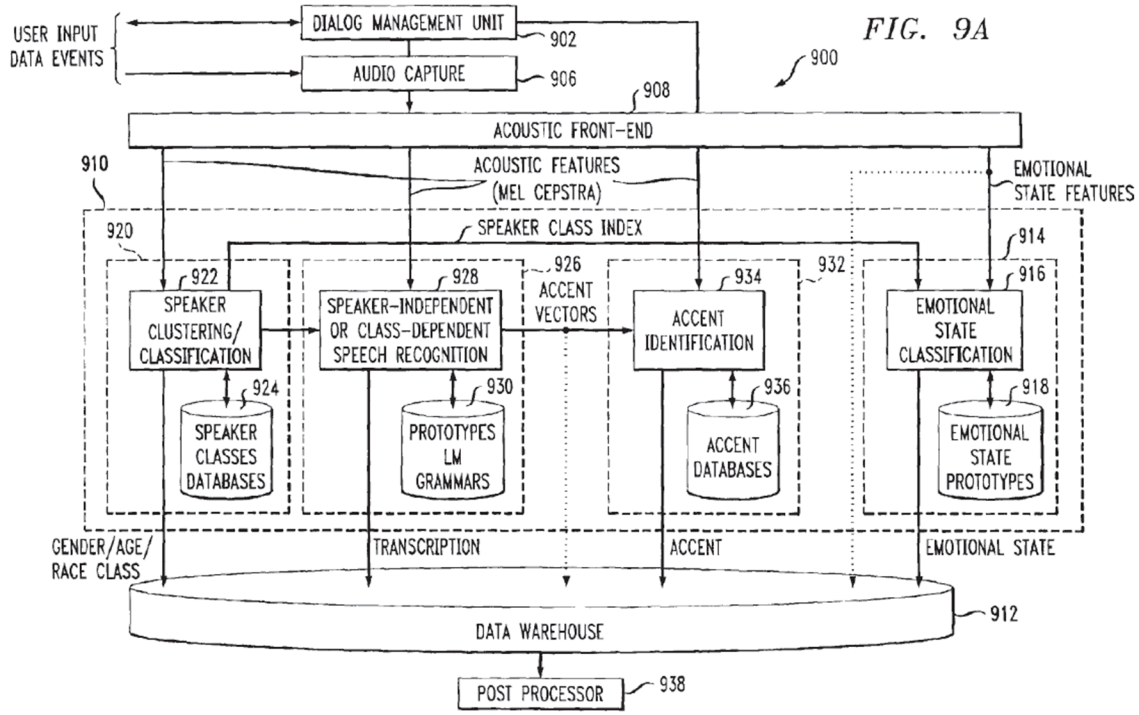
Claim 7 of the '957 Patent	**Exemplary Citations to Maes**
	or herein, for the reasons identified herein or in the Cover Pleading. A person of ordinary skill in the art would have been motivated to combine Maes with any of the other prior art references identified in the Cover Pleading or herein with a reasonable expectation of success because each of these references contains familiar elements within the similar fields of endeavor to address similar known problems, according to known methods which would yield predictable results.
[a] generating a context stack comprising context information that corresponds to a plurality of prior utterances, wherein the context stack includes a plurality of context entries;	Maes discloses "generating a context stack comprising context information that corresponds to a plurality of prior utterances, wherein the context stack includes a plurality of context entries[.]" *E.g.*, Maes at Fig. 2:

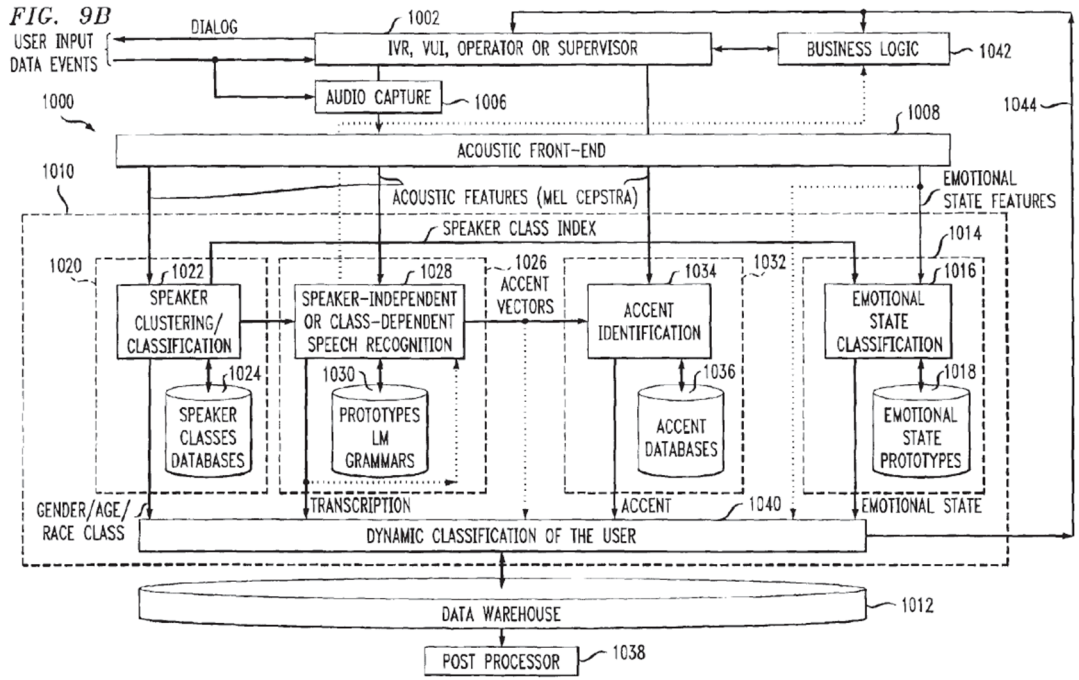
FIG. 2

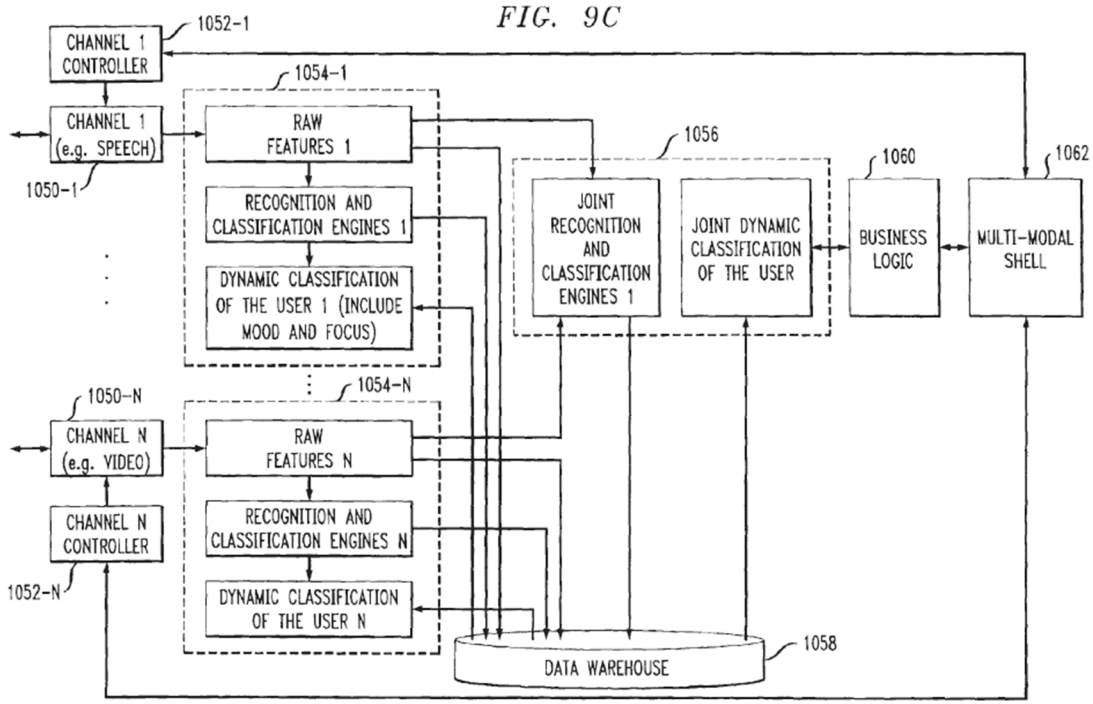

200
START
OBTAIN RAW MULTI-MODAL INPUT DATA FROM MULTI-MODAL DATA SOURCES
202
I/O MANAGER ABSTRACTS RAW INPUT DATA INTO EVENTS
204
ONE OR MORE RECOGNITION ENGINES PROCESS EVENT, IF NECESSARY, BASED ON THE NATURE OF THE INPUT DATA
206
208
STORE EVENTS ON CONTEXT STACK
DIALOG MANAGER ATTEMPTS TO DETERMINE INTENT OF USER BASED ON PROCESSED ABSTRACT EVENT AND CONTEXT STACK HISTORY
210
212
NO
INTENT KNOWN ?
YES
214
DIALOG MANAGER CAUSES GENERATION OF OUTPUT TO USER TO OBTAIN FURTHER INPUT DATA
216
NO
CONFIRM INTENT ?
YES
EXECUTE PROCESSED EVENT OR OTHER ACTION
218

Claim 7 of the '957 Patent	Exemplary Citations to Maes
	Maes at Fig. 3:  Maes at Fig 8B:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 9A:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 9B:

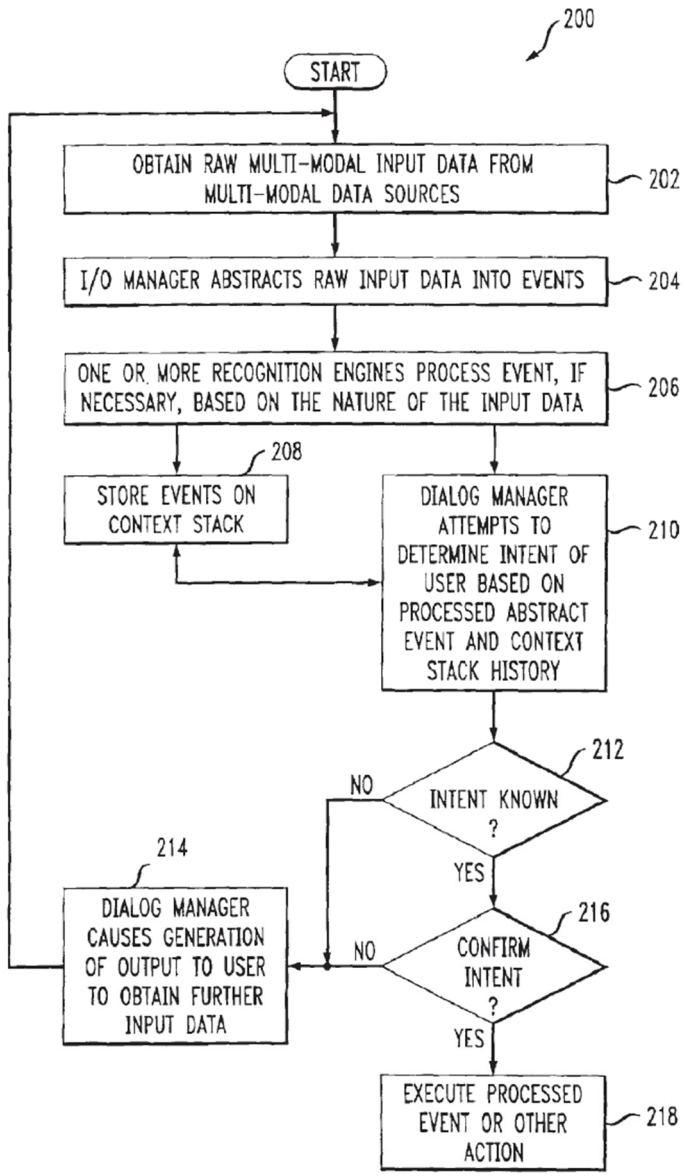
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 9C:

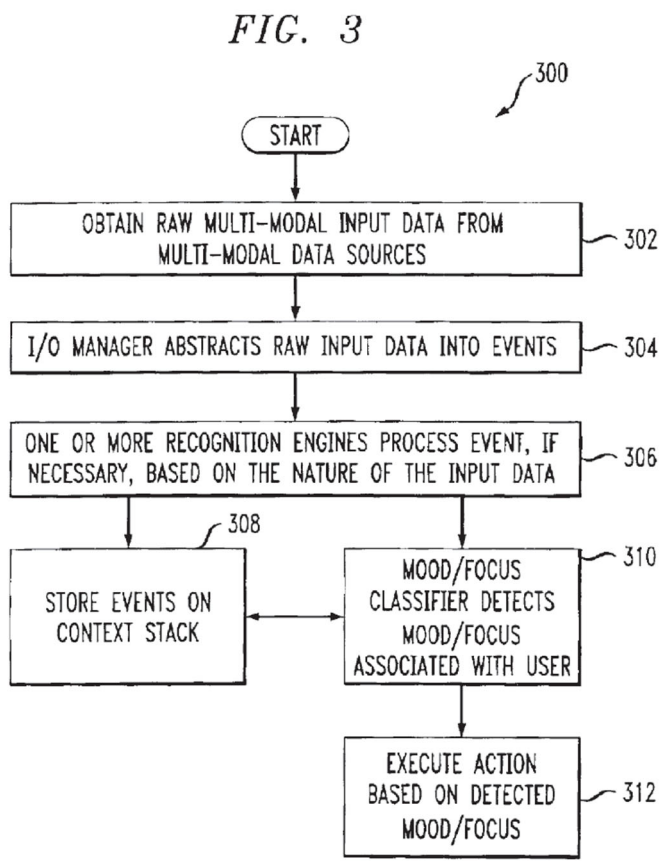
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	FIG. 9C  Maes at 5:15-19: This may also include disambiguating an I/O (input/output) event generated previously and stored in a context manager/history stack (e.g., if a beeper rang and the user asked "turn it off," the term "it" can be disambiguated). Techniques for accomplishing this will be explained below. Maes at 7:33-45: In step 208, the recognized events, as well as the events that do not need to be recognized, are stored in a storage unit referred to as the context stack 20. The context stack is used to create a

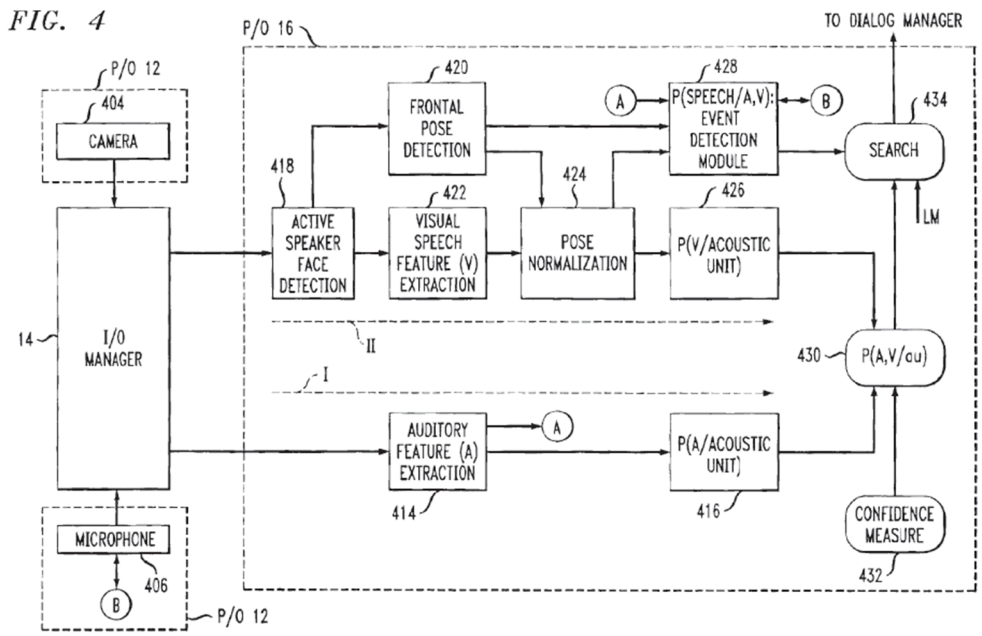
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	history of interaction between the user and the system so as to assist the dialog manager 18 in making referential ambiguity resolution determinations when determining the user's intent. Next, in step 210, the system 10 attempts to determine the user intent based on the current event and the historical interaction information stored in the context stack and then determine and execute one or more application programs that effectuate the user's intention and/or react to the user activity. Maes at 37:17-50: The task dispatcher/manager 821 dispatches and coordinates different tasks and processes that are spawned (by the user and machine) on local and networked conventional and conversational resources. The meta-information manager 822 manages the meta-information associated with the system via a meta-information repository 818. The meta-information manager 822 and repository 818 collect all the information typically assumed known in a conversational interaction but not available at the level of the current conversation. Examples are a-priori knowledge, cultural, educational assumptions and persistent information, past request, references, information about the user, the application, news, etc. It is typically the information that needs to be preserved and persist beyond the length/life of the conversational history/context and the information that is expected to be common knowledge for the conversation and, therefore, has never been defined during the current and possible past conversational interactions. Also, shortcuts to commands, resources and macros, etc., are managed by the meta-information manager 822 and stored in the meta-information repository 818. In addition, the meta-information repository 818 includes a user-usage log based on user identity. It is to be appreciated that services such as conversational help and assistance, as well as some dialog prompts (introduction, questions, feedback, etc.) provided by the CVM system can be tailored based on the usage history of the user as stored in the meta-information repository 818 and associated with the application. If a user has been previously interacting with a given application, an explanation can be reduced assuming that it is familiar to the user. Similarly, if a user commits many errors, the explanations can be more complex, as multiple errors are interpreted as user uncertainty, unfamiliarity, or incomprehension/misunderstanding of the application or function.

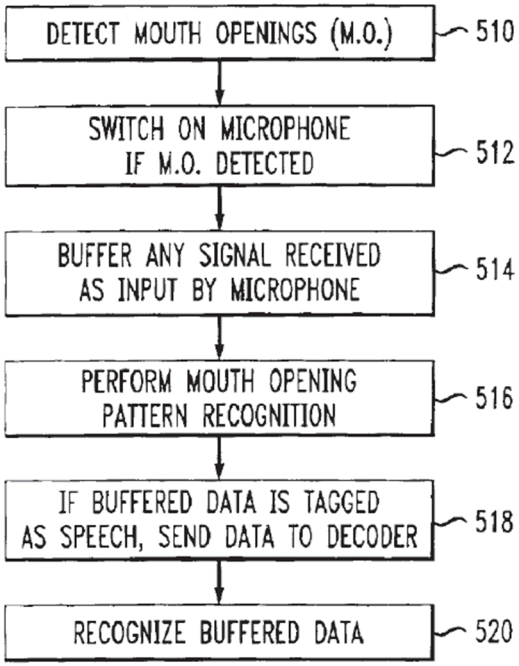
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Maes at 37:51-38:2: A context stack 817 is managed by the dialog manager 819, possibly through a context manager that interacts with the dialog manager and arbitrator. It is to be understood that the context stack 817 may be implemented as part of the context stack 20 of FIG. 1. The context stack 817 comprises all the information associated with an application. Such information includes all the variable, states, input, output and queries to the backend that are performed in the context of the dialog and any extraneous event that occurs during the dialog. The context stack is associated with the organized/sorted context corresponding to each active dialog (or deferred dialog-agents/memorization). A global history 816 is included in the CVM system and includes information that is stored beyond the context of each application. The global history stores, for example, the information that is associated with all the applications and actions taking during a conversational session (i.e., the history of the dialog between user and machine for a current session or from when the machine was activated). To the extent it is found that Maes does not expressly disclose this limitation, the limitation is inherent. And to the extent it is found that Maes does not anticipate this claim, Maes renders it obvious, standing alone, in combination with the knowledge of a person of ordinary skill in the art, and/or in combination with one or more other prior art references identified in the Cover Pleading or herein, for the reasons identified herein or in the Cover Pleading. A person of ordinary skill in the art would have been motivated to combine Maes with any of the other prior art references identified in the Cover Pleading or herein with a reasonable expectation of success because each of these references contains familiar elements within the similar fields of endeavor to address similar known problems, according to known methods which would yield predictable results.
[b] receiving, by one or more computer processors, the natural language utterance, wherein the natural language utterance is associated with a command or is associated with a request;	Maes discloses “receiving, by one or more computer processors, the natural language utterance, wherein the natural language utterance is associated with a command or is associated with a request[.]” *E.g.*, Maes at Fig. 2:

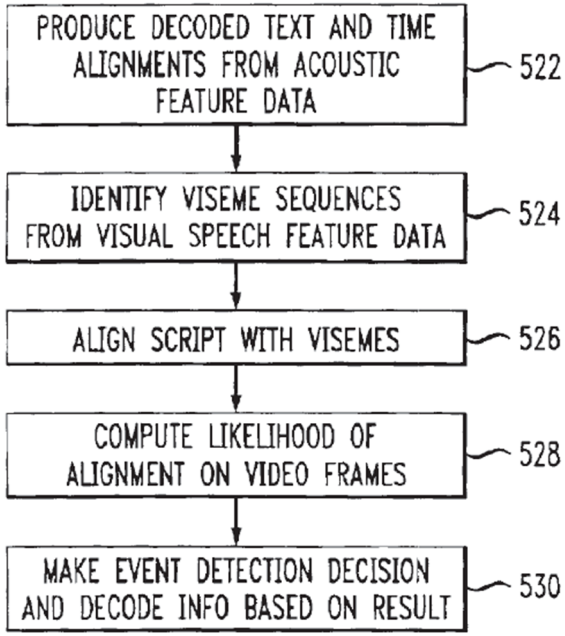
FIG. 2

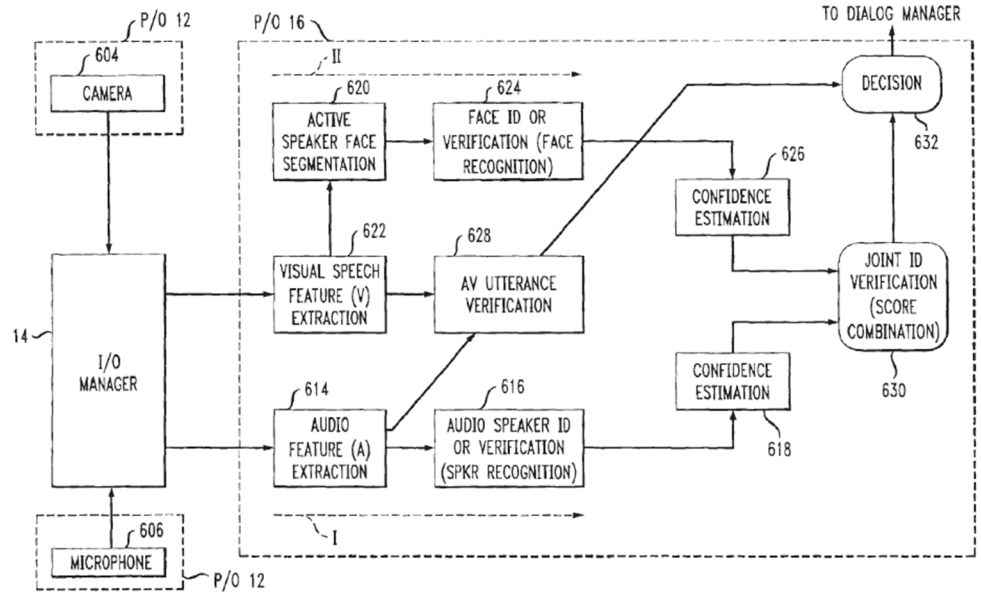

200
START
OBTAIN RAW MULTI-MODAL INPUT DATA FROM MULTI-MODAL DATA SOURCES
202
I/O MANAGER ABSTRACTS RAW INPUT DATA INTO EVENTS
204
ONE OR MORE RECOGNITION ENGINES PROCESS EVENT, IF NECESSARY, BASED ON THE NATURE OF THE INPUT DATA
206
208
STORE EVENTS ON CONTEXT STACK
DIALOG MANAGER ATTEMPTS TO DETERMINE INTENT OF USER BASED ON PROCESSED ABSTRACT EVENT AND CONTEXT STACK HISTORY
210
212
NO
INTENT KNOWN ?
YES
214
DIALOG MANAGER CAUSES GENERATION OF OUTPUT TO USER TO OBTAIN FURTHER INPUT DATA
216
NO
CONFIRM INTENT ?
YES
EXECUTE PROCESSED EVENT OR OTHER ACTION
218

Claim 7 of the '957 Patent	Exemplary Citations to Maes
	Maes at Fig. 3:  Maes at Fig. 4:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 5C:

Claim 7 of the '957 Patent	Exemplary Citations to Maes
	FIG. 5C  Maes at 5D:

Claim 7 of the '957 Patent	Exemplary Citations to Maes
	FIG. 5D  Maes at Fig. 6:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	FIG. 6  Maes at 4:62-5:5: Thus, the system 10 must be able to perform user reference resolution, e.g., the system may receive the spoken utterance, “call my office,” but unless the system can resolve which occupant made this statement, it will not know which office phone number to direct an associated cellular telephone to call. The system 10 therefore performs referential ambiguity resolution with respect to multiple users by taking both audio input data and image data input and processing it to make a user resolution determination. This may include detecting speech activity and/or the identity of the user based on both audio and image cues. Maes at 5:7-15:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Similarly, a user may say to the system, “turn that off,” but without device reference resolution, the system would not know which associated device to direct to be turned off. The system 10 therefore performs referential ambiguity resolution with respect to multiple devices by taking both audio input data and image data input and processing it to make a device resolution determination. This may include detecting the speaker's head pose using gross spatial resolution of the direction being addressed, or body pose (e.g., pointing). Maes at 5:57-6:20: First, in step 202, raw multi-modal input data is obtained from multi-modal data sources associated with the system. In terms of the computing system 10 in FIG. 1, such sources are represented by I/O subsystem 12. As mentioned above, the data input portion of the subsystem may comprise one or more cameras or sensors for capturing video input data representing the environment in which the system (or, at least, the I/O subsystem) is deployed. The cameras/sensors may be capable of capturing not only visible image data (images in the visible electromagnetic spectrum), but also IR (near, mid and/or far field IR video) and/or RF image data. Of course, in systems with more than one camera, different mixes of cameras/sensors may be employed, e.g., system having one or more video cameras, one or more IR sensors and/or one or more RF sensors. In addition to the one or more cameras, the I/O subsystem 12 may comprise one or more microphones for capturing audio input data from the environment in which the system is deployed. Further, the I/O subsystem may also include an analog-to-digital converter which converts the electrical signal generated by a microphone into a digital signal representative of speech uttered or other sounds that are captured. Further, the subsystem may sample the speech signal and partition the signal into overlapping frames so that each frame is discretely processed by the remainder of the system. Thus, referring to the vehicle example above, it is to be understood that the cameras and microphones may be strategically placed throughout the vehicle in order to attempt to fully

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	capture all visual activity and audio activity that may be necessary for the system to make ambiguity resolution determinations. Maes at 6:21-28: Still further, the I/O subsystem 12 may also comprise other typical input devices for obtaining user input, e.g., GUI-based devices such as a keyboard, a mouse, etc., and/or other devices such as a stylus and digitizer pad for capturing electronic handwriting, etc. It is to be understood that one of ordinary skill in the art will realize other user interfaces and devices that may be included for capturing user activity. Maes at 10:8-19: The focus/mood classifier 22 receives either the events directly from the I/O manager 14 or, if necessary depending on the nature of the event, the classifier receives the recognized events from the one or more recognition engines 16. For instance, in the vehicle example, the focus/mood classifier may receive visual events indicating the position of the user's eyes and/or head as well as audio events indicating sounds the user may be making (e.g., snoring). Using these events, as well as past information stored on the context stack, the classifier makes the focus detection and/or mood classification determination. Results of such determinations may also be stored on the context stack. Maes at 10:47-11:1: Referring now to FIG. 4, a block diagram illustrates a preferred embodiment of an audio-visual speech recognition module that may be employed as one of the recognition modules of FIG. 1 to perform speech recognition using multi-modal input data received in accordance with the invention. It is to be appreciated that such an audio-visual speech recognition module is disclosed in the above-referenced U.S. patent application identified as Ser. No. 09/369,707, filed on Aug. 6, 1999 and entitled “Methods and Apparatus for Audio-visual Speech Detection and Recognition.” A description of one of the embodiments of such an audio-visual speech recognition module for

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	use in a preferred embodiment of the multi-modal conversational computing system of the invention is provided below in this section. However, it is to be appreciated that other mechanisms for performing speech recognition may be employed. This particular illustrative embodiment, as will be explained, depicts audio-visual recognition using a decision fusion approach. It is to be appreciated that one of the advantages that the audio-visual speech recognition module described herein provides is the ability to process arbitrary content video. Maes at 11:32-54: It is to be understood that the video signal captured by the camera 404 can be of any particular type. As mentioned, the face and pose detection techniques may process images of any wavelength such as, e.g., visible and/or non-visible electromagnetic spectrum images. By way of example only, this may include infrared (IR) images (e.g., near, mid and far field IR video) and radio frequency (RF) images. Accordingly, the module may perform audio-visual speech detection and recognition techniques in poor lighting conditions, changing lighting conditions, or in environments without light. For example, the system may be installed in an automobile or some other form of vehicle and capable of capturing IR images so that improved speech recognition may be performed. Because video information (i.e., including visible and/or non-visible electromagnetic spectrum images) is used in the speech recognition process, the system is less susceptible to recognition errors due to noisy conditions, which significantly hamper conventional recognition systems that use only audio information. In addition, due to the methodologies for processing the visual information described herein, the module provides the capability to perform accurate LVCSR (large vocabulary continuous speech recognition). Maes at 12:49-60: Referring now to the video signal path II of FIG. 4, the methodologies of processing visual information will now be explained. The audio-visual speech recognition module (denoted in FIG. 4 as part of block 16 from FIG. 1) includes an active speaker face detection module 418. The

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	active speaker face detection module 418 receives video input camera 404. It is to be appreciated that speaker face detection can also be performed directly in the compressed data domain and/or from audio and video information rather than just from video information. In any case, module 418 generally locates and tracks the speaker's face and facial features within the arbitrary video background. Maes at 18:42-63: The extracted visual speech feature vectors are then normalized in block 424 with respect to the frontal pose estimates generated by the detection module 420. The normalized visual speech feature vectors are then provided to a probability module 426. Similar to the probability module 416 in the audio information path which labels the acoustic feature vectors with one or more phonemes, the probability module 426 labels the extracted visual speech vectors with one or more previously stored phonemes. Again, each phoneme associated with one or more visual speech feature vectors has a probability associated therewith indicating the likelihood that it was that particular acoustic unit that was spoken in the video segment being considered. Thus, the probability module yields likelihood scores for each considered phoneme in the form of the probability that, given a particular phoneme or acoustic unit (au), the acoustic unit represents the uttered speech characterized by one or more visual speech feature vectors V or, in other words, P(V|acoustic unit). Alternatively, the visual speech feature vectors may be labeled with visemes which, as previously mentioned, are visual phonemes or canonical mouth shapes that accompany speech utterances. Maes at 20:21-21:30: In a preferred embodiment, the audio-visual speech recognition module of FIG. 4 also includes an event detection module 428. As previously mentioned, one problem of conventional speech recognition systems is there inability to discriminate between extraneous audible activity, e.g., background noise or background speech not intended to be decoded, and speech that is indeed intended to be decoded. This causes such problems as misfiring of the system and “junk” recognition. According to various embodiments, the module may use information from the video

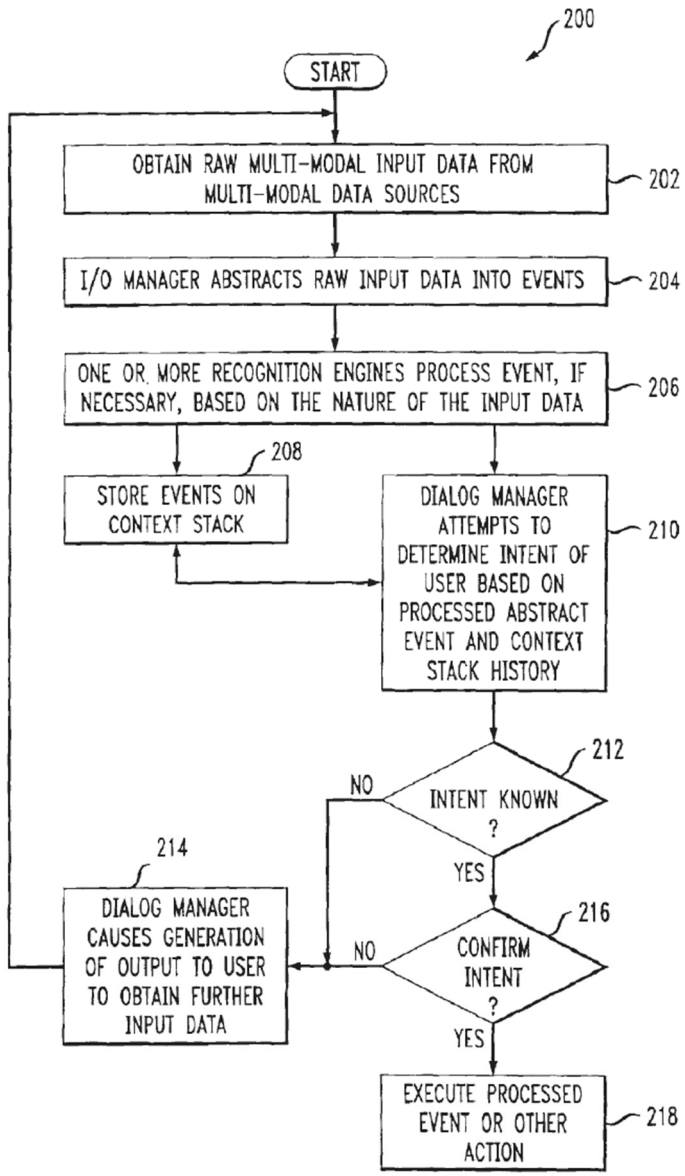
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	path only, information from the audio path only, or information from both paths simultaneously to decide whether or not to decode information. This is accomplished via the event detection module 428. It is to be understood that “event detection” refers to the determination of whether or not an actual speech event that is intended to be decoded is occurring or is going to occur. Based on the output of the event detection module, microphone 406 or the search engine 434 may be enabled/disabled. Note that if no face is detected, then the audio can be processed to make decisions. Maes at 31:43-32:45: A CVM (or operating system) based on the conversational computing paradigm described herein allows a computer or any other interactive device to converse with a user. The CVM further allows the user to run multiple tasks on a machine regardless if the machine has no display or GUI capabilities, nor any keyboard, pen or pointing device. Indeed, the user can manage these tasks like a conversation and bring a task or multiple simultaneous tasks, to closure. To manage tasks like a conversation, the CVM affords the capability of relying on mixed initiatives, contexts and advanced levels of abstraction, to perform its various functions. Mixed initiative or free flow navigation allows a user to naturally complete, modify, or correct a request via dialog with the system. Mixed initiative also implies that the CVM can actively help (take the initiative to help) and coach a user through a task, especially in speech-enabled applications, wherein the mixed initiative capability is a natural way of compensating for a display-less system or system with limited display capabilities. In general, the CVM complements conventional interfaces and user input/output rather than replacing them. This is the notion of “multi-modality” whereby speech, and video as described above, may be used in parallel with a mouse, keyboard, and other input devices such as a pen. Conventional interfaces can be replaced when device limitations constrain the implementation of certain interfaces. In addition, the ubiquity and uniformity of the resulting interface across devices, tiers and services is an additional mandatory characteristic. It is to be understood that a CVM system can, to a large extent, function with conventional input and/or output media. Indeed, a computer with classical keyboard inputs and pointing devices coupled with a traditional monitor display can profit significantly by utilizing the CVM. One example is described in U.S. patent application identified as U.S. Ser. No. 09/507,526 filed on Feb. 18, 2000

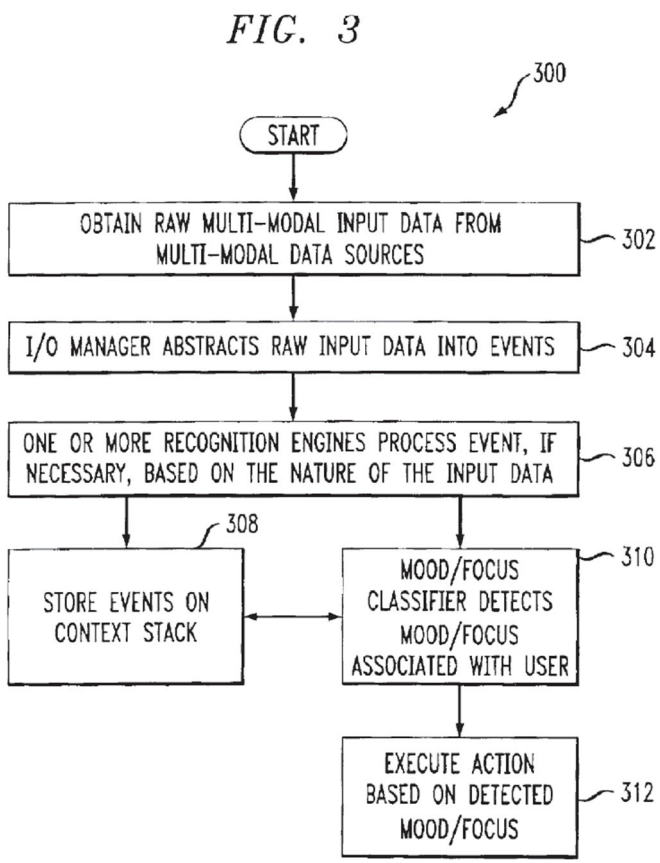
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	and entitled “Multi-Modal Shell” which claims priority to U.S. provisional patent application identified as U.S. Ser. No. 60/128,081 filed on Apr. 7, 1999 and U.S. provisional patent application identified by Ser. No. 60/158,777 filed on Oct. 12, 1999, the disclosures of which are incorporated by reference herein (which describes a method for constructing a true multi-modal application with tight synchronization between a GUI modality and a speech modality). In other words, even users who do not want to talk to their computer can also realize a dramatic positive change to their interaction with the CVM enabled machine. Referring now to FIG. 8A, a block diagram illustrates a CVM system according to a preferred embodiment, which may be implemented on a client device or a server. In terms of the vehicle example above, this means that the components of the system 10 may be located locally (in the vehicle), remotely (e.g., connected wirelessly to the vehicle), or some combination thereof. In general, the CVM provides a universal coordinated multi-modal conversational user interface (CUI) 780. The “multi-modality” aspect of the CUI implies that various I/O resources such as voice, keyboard, pen, and pointing device (mouse), keypads, touch screens, etc., and video as described above, can be used in conjunction with the CVM platform. The “universality” aspect of the CUI implies that the CVM system provides the same UI (user interface) to a user whether the CVM is implemented in connection with a desktop computer, a PDA with limited display capabilities, or with a phone where no display is provided. In other words, universality implies that the CVM system can appropriately handle the UI of devices with capabilities ranging from speech only to multi-modal, i.e., speech + GUI, to purely GUI. As per the present invention, the system may be extended to include video input data as well. Therefore, the universal CUI provides the same UI for all user interactions, regardless of the access modality. Maes at 39:14-43: FIGS. 9A and 9B will be used to generally describe mood/focus classification techniques that may be employed in the mood/focus classifier 22 (FIG. 1) with respect to speech-based event data. However, the extended application to include the modality associated with video-based event data will be illustrated in the context of FIG. 9C where it is shown that these classification techniques can be easily applied to multi-modal input.

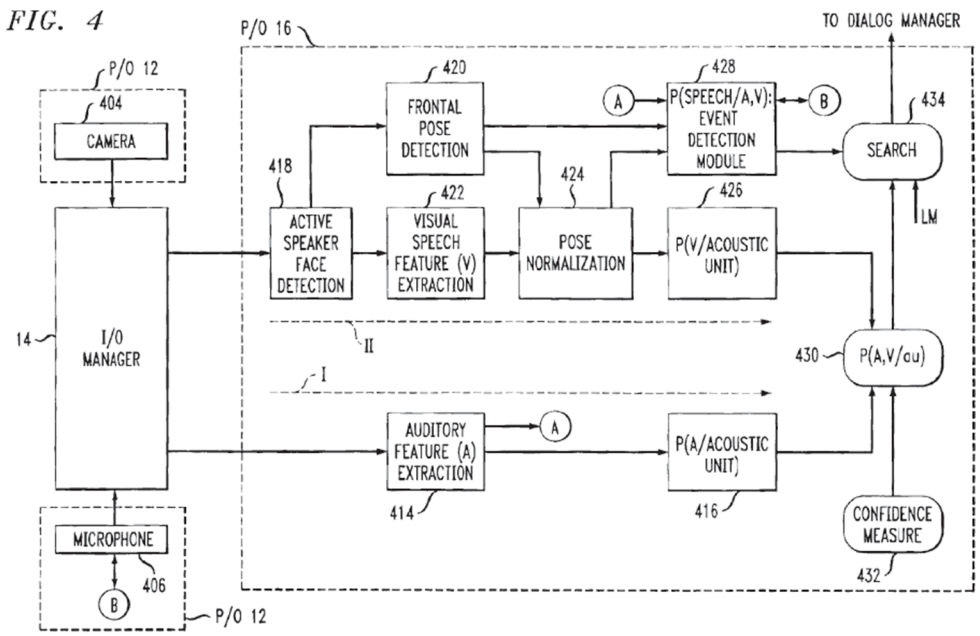
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	FIG. 9A depicts an apparatus for collecting data associated with a voice of a user, in accordance with the present invention. The apparatus is designated generally as 900. The apparatus includes a dialog management unit 902 which conducts a conversation with the user. It is to be understood that the user-provided input data events are preferably provided to the system 900 via the I/O manager 14 of FIG. 1. Apparatus 900 further includes an audio capture module 906 which is coupled to the dialog management unit 902 and which captures a speech waveform associated with utterances spoken by the user 904 during the conversation. While shown for ease of explanation in FIG. 9A, the audio capture unit 906 may be part of the I/O subsystem 12 of FIG. 1. In which case, the captured input data is passed onto system 900 via the I/O manager 14. As used herein, a conversation should be broadly understood to include any interaction, between a first human and either a second human, a machine, or a combination thereof, which includes at least some speech. Again, based on the above described teachings of the multi-modal system 10 of the invention, the mood classification (focus detection) system 900 may be extended to process video in a similar manner. Maes at 41:35-44: Processing module 910 can further include a speech recognizor 926 which can, in turn, itself include a speech recognition module 928 and a speech prototype, language model and grammar database 930. Speech recognizor 926 can be part of the dialog management unit 902 or, for example, a separate element within the implementation of processing module 910. Yet further, processing module 910 can include an accent identifier 932, which in turn includes an accent identification module 934 and an accent database 936. To the extent it is found that Maes does not expressly disclose this limitation, the limitation is inherent. And to the extent it is found that Maes does not anticipate this claim, Maes renders it obvious, standing alone, in combination with the knowledge of a person of ordinary skill in the art, and/or in combination with one or more other prior art references identified in the Cover Pleading or herein, for the reasons identified herein or in the Cover Pleading. A person of ordinary skill in the art would have been motivated to combine Maes with any of the other prior art references

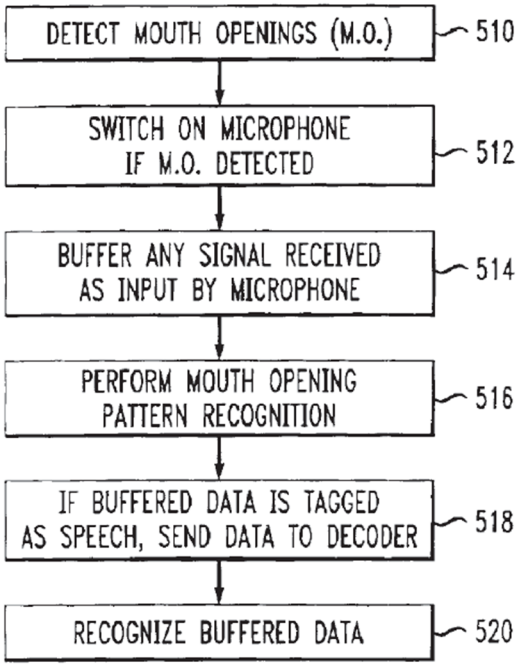
Claim 7 of the '957 Patent	**Exemplary Citations to Maes**
	identified in the Cover Pleading or herein with a reasonable expectation of success because each of these references contains familiar elements within the similar fields of endeavor to address similar known problems, according to known methods which would yield predictable results.
[c] determining one or more words of the natural language utterance by performing speech recognition on the natural language utterance;	Maes discloses "determining one or more words of the natural language utterance by performing speech recognition on the natural language utterance[.]" *E.g.*, Maes at Fig. 2:

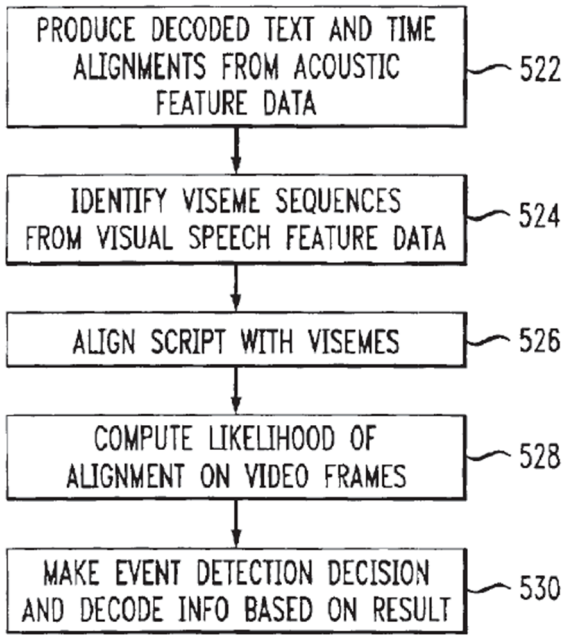
FIG. 2

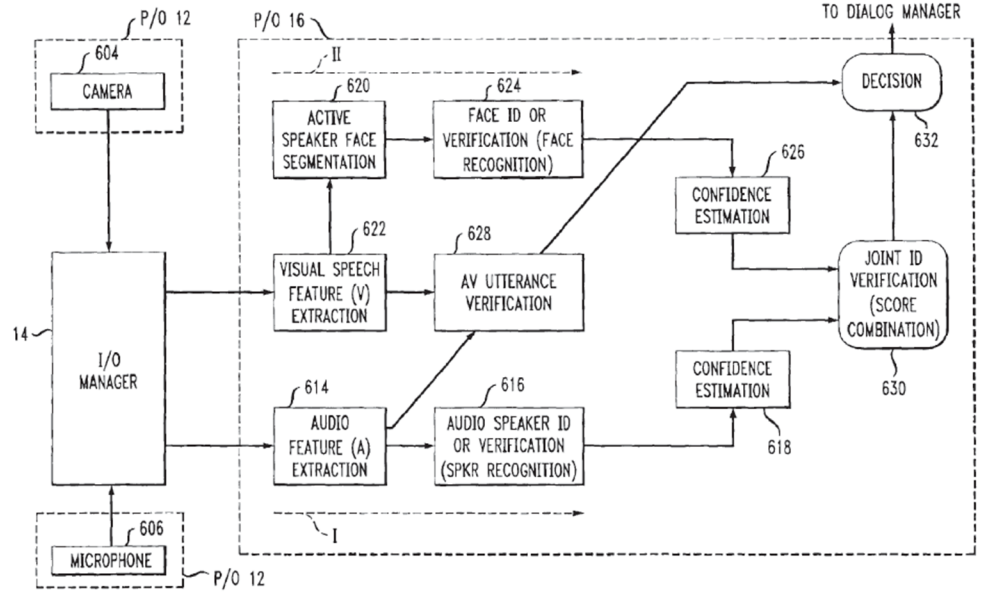

200
START
OBTAIN RAW MULTI-MODAL INPUT DATA FROM MULTI-MODAL DATA SOURCES
202
I/O MANAGER ABSTRACTS RAW INPUT DATA INTO EVENTS
204
ONE OR MORE RECOGNITION ENGINES PROCESS EVENT, IF NECESSARY, BASED ON THE NATURE OF THE INPUT DATA
206
208
STORE EVENTS ON CONTEXT STACK
DIALOG MANAGER ATTEMPTS TO DETERMINE INTENT OF USER BASED ON PROCESSED ABSTRACT EVENT AND CONTEXT STACK HISTORY
210
212
NO
INTENT KNOWN ?
YES
214
DIALOG MANAGER CAUSES GENERATION OF OUTPUT TO USER TO OBTAIN FURTHER INPUT DATA
216
NO
CONFIRM INTENT ?
YES
EXECUTE PROCESSED EVENT OR OTHER ACTION
218

Claim 7 of the '957 Patent	Exemplary Citations to Maes
	Maes at Fig. 3:  Maes at Fig. 4:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 5C:

Claim 7 of the '957 Patent	Exemplary Citations to Maes
	FIG. 5C  Maes at 5D:

Claim 7 of the '957 Patent	Exemplary Citations to Maes
	FIG. 5D  Maes at Fig. 6:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	FIG. 6  Maes at 6:5-15: In addition to the one or more cameras, the I/O subsystem 12 may comprise one or more microphones for capturing audio input data from the environment in which the system is deployed. Further, the I/O subsystem may also include an analog-to-digital converter which converts the electrical signal generated by a microphone into a digital signal representative of speech uttered or other sounds that are captured. Further, the subsystem may sample the speech signal and partition the signal into overlapping frames so that each frame is discretely processed by the remainder of the system. Maes at 6:29-38:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Next, in step 204, the raw multi-modal input data is abstracted into one or more events. In terms of the computing system 10 in FIG. 1, the data abstraction is performed by the I/O manager 14. The I/O manager receives the raw multi-modal data and abstracts the data into a form that represents one or more events, e.g., a spoken utterance, a visual gesture, etc. As is known, a data abstraction operation may involve generalizing details associated with all or portions of the input data so as to yield a more generalized representation of the data for use in further operations. Maes at 6:39-7:20: In step 206, the abstracted data or event is then sent by the I/O manager 14 to one or more recognition engines 16 in order to have the event recognized, if necessary. That is, depending on the nature of the event, one or more recognition engines may be used to recognize the event. For example, if the event is some form of spoken utterance wherein the microphone picks up the audible portion of the utterance and a camera picks up the visual portion (e.g., lip movement) of the utterance, the event may be sent to an audio-visual speech recognition engine to have the utterance recognized using both the audio input and the video input associated with the speech. Alternatively, or in addition, the event may be sent to an audio-visual speaker recognition engine to have the speaker of the utterance identified, verified and/or authenticated. Also, both speech recognition and speaker recognition can be combined on the same utterance. If the event is some form of user gesture picked up by a camera, the event may be sent to a gesture recognition engine for recognition. Again, depending on the types of user interfaces provided by the system, the event may comprise handwritten input provided by the user such that one of the recognition engines may be a handwriting recognition engine. In the case of more typical GUI-based input (e.g., keyboard, mouse, etc.), the data may not necessarily need to be recognized since the data is already identifiable without recognition operations. An audio-visual speech recognition module that may be employed as one of the recognition engines 16 is disclosed in U.S. patent application identified as Ser. No. 09/369,707, filed on Aug. 6, 1999 and entitled “Methods and Apparatus for Audio-visual Speech Detection and Recognition,” the disclosure of which is incorporated by reference herein. A description of such an

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	audio-visual speech recognition system will be provided below. An audio-visual speaker recognition module that may be employed as one of the recognition engines 16 is disclosed in U.S. patent application identified as Ser. No. 09/369,706, filed on Aug. 6, 1999 and entitled “Methods And Apparatus for Audio-Visual Speaker Recognition and Utterance Verification,” the disclosure of which is incorporated by reference herein. A description of such an audio-visual speaker recognition system will be provided below. It is to be appreciated that gesture recognition (e.g., body, arms and/or hand movement, etc., that a user employs to passively or actively give instruction to the system) and focus recognition (e.g., direction of face and eyes of a user) may be performed using the recognition modules described in the above-referenced patent applications. Maes at 7:23-33: It is to be appreciated that two, more or even all of the input modes described herein may be synchronized via the techniques disclosed in U.S. patent application identified as Ser. No. 09/507,526 filed on Feb. 18, 2000 and entitled “Systems and Method for Synchronizing Multi-modal Interactions,” which claims priority to U.S. provisional patent application identified as U.S. Ser. No. 60/128,081 filed on Apr. 7, 1999 and U.S. provisional patent application identified by Ser. No. 60/158,777 filed on Oct. 12, 1999, the disclosures of which are incorporated by reference herein. Maes at 11:64-12:28: The module includes an auditory feature extractor 414. The feature extractor 414 receives an audio or speech signal and, as is known in the art, extracts spectral features from the signal at regular intervals. The spectral features are in the form of acoustic feature vectors (signals) which are then passed on to a probability module 416. Before acoustic vectors are extracted, the speech signal may be sampled at a rate of 16 kilohertz (kHz). A frame may consist of a segment of speech having a 25 millisecond (msec) duration. In such an arrangement, the extraction process preferably produces 24 dimensional acoustic cepstral vectors via the process described below. Frames are advanced every 10 msec to obtain succeeding acoustic vectors. Note that other acoustic front-ends with other frame sizes and sampling rates/signal bandwidths can also be employed.

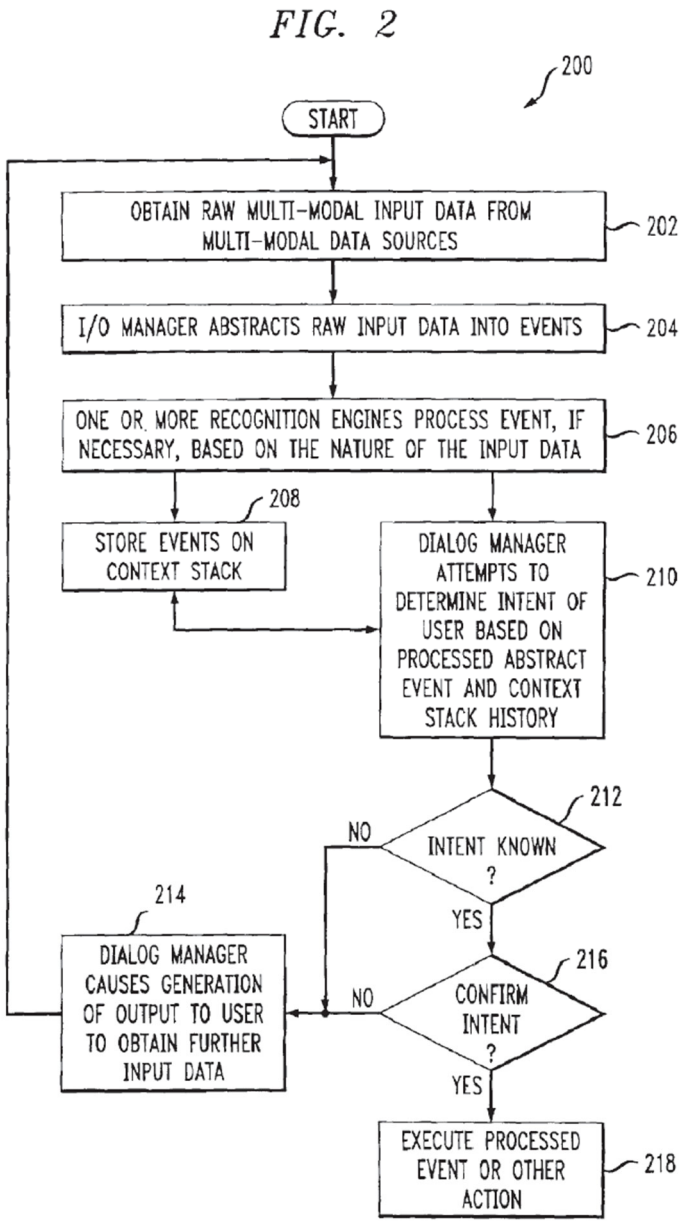
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	First, in accordance with a preferred acoustic feature extraction process, magnitudes of discrete Fourier transforms of samples of speech data in a frame are considered in a logarithmically warped frequency scale. Next, these amplitude values themselves are transformed to a logarithmic scale. The latter two steps are motivated by a logarithmic sensitivity of human hearing to frequency and amplitude. Subsequently, a rotation in the form of discrete cosine transform is applied. One way to capture the dynamics is to use the delta (first-difference) and the delta-delta (second-order differences) information. An alternative way to capture dynamic information is to append a set of (e.g., four) preceding and succeeding vectors to the vector under consideration and then project the vector to a lower dimensional space, which is chosen to have the most discrimination. The latter procedure is known as Linear Discriminant Analysis (LDA) and is well known in the art. Maes at 12:29-48: After the acoustic feature vectors, denoted in FIG. 4. by the letter A, are extracted, the probability module labels the extracted vectors with one or more previously stored phonemes which, as is known in the art, are sub-phonetic or acoustic units of speech. The module may also work with lefemes, which are portions of phones in a given context. Each phoneme associated with one or more feature vectors has a probability associated therewith indicating the likelihood that it was that particular acoustic unit that was spoken. Thus, the probability module yields likelihood scores for each considered phoneme in the form of the probability that, given a particular phoneme or acoustic unit (au), the acoustic unit represents the uttered speech characterized by one or more acoustic feature vectors A or, in other words, P(A|acoustic unit). It is to be appreciated that the processing performed in blocks 414 and 416 may be accomplished via any conventional acoustic information recognition system capable of extracting and labeling acoustic feature vectors, e.g., Lawrence Rabiner, Biing-Hwang Juang, “Fundamentals of Speech Recognition,” Prentice Hall, 1993. Maes at 18:42-63:

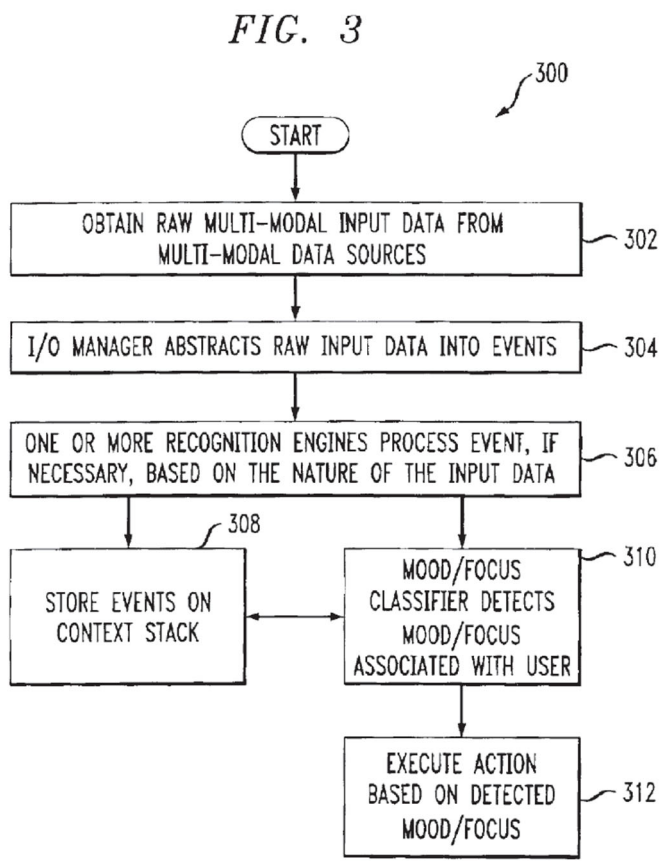
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	The extracted visual speech feature vectors are then normalized in block 424 with respect to the frontal pose estimates generated by the detection module 420. The normalized visual speech feature vectors are then provided to a probability module 426. Similar to the probability module 416 in the audio information path which labels the acoustic feature vectors with one or more phonemes, the probability module 426 labels the extracted visual speech vectors with one or more previously stored phonemes. Again, each phoneme associated with one or more visual speech feature vectors has a probability associated therewith indicating the likelihood that it was that particular acoustic unit that was spoken in the video segment being considered. Thus, the probability module yields likelihood scores for each considered phoneme in the form of the probability that, given a particular phoneme or acoustic unit (au), the acoustic unit represents the uttered speech characterized by one or more visual speech feature vectors V or, in other words, P(V|acoustic unit). Alternatively, the visual speech feature vectors may be labeled with visemes which, as previously mentioned, are visual phonemes or canonical mouth shapes that accompany speech utterances. Maes at 23:10-23: The module includes an auditory feature extractor 614. The feature extractor 614 receives an audio or speech signal and, as is known in the art, extracts spectral features from the signal at regular intervals. The spectral features are in the form of acoustic feature vectors (signals) which are then passed on to an audio speaker recognition module 616. Before acoustic vectors are extracted, the speech signal may be sampled at a rate of 16 kilohertz (kHz). A frame may consist of a segment of speech having a 25 millisecond (msec) duration. In such an arrangement, the extraction process preferably produces 24 dimensional acoustic cepstral vectors via the process described below. Frames are advanced every 10 msec to obtain succeeding acoustic vectors. Of course, other front-ends may be employed. Maes at 36:8-48: The core CVM kernel layer 802 comprises programming layers such as a conversational application and behavior/service manager layer 815, a conversational dialog manager (arbitrator)

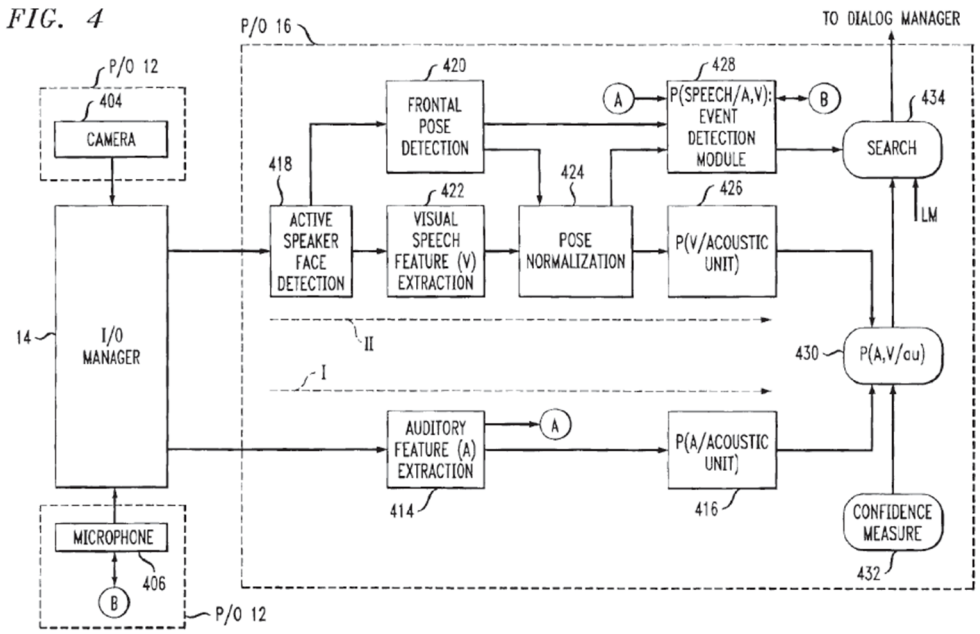
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	layer 819, a conversational resource manager layer 820, a task/dispatcher manager 821 and a meta-information manager 822, which provide the core functions of the CVM layer 802. It is to be understood that these components may be implemented as part of the dialog manager 18 of FIG. 1. The conversational application and behavior/service manager layer 815 comprises functions for managing the conventional and conversationally aware applications 800 and 801. Such management functions include, for example, keeping track of which applications are registered (both local and network-distributed), what are the dialog interfaces (if any) of the applications, and what is the state of each application. In addition, the conversational application and services/behavior manager 815 initiates all the tasks associated with any specific service or behavior provided by the CVM system. The conversational services and behaviors are all the behaviors and features of a conversational UI that the user may expect to find in the applications and interactions, as well as the features that an application developer may expect to be able to access via APIs (without having to implement with the development of the application). Examples of the conversational services and behavior provided by the CVM kernel 802 include, but are not limited to, conversational categorization and meta-information, conversational object, resource and file management, conversational search, conversational selection, conversational customization, conversational security, conversational help, conversational prioritization, conversational resource management, output formatting and presentation, summarization, conversational delayed actions/agents/memorization, conversational logic, and coordinated interfaces and devices. Such services are provided through API calls via the conversational application API Layer 803. The conversational application and behavior/services manager 815 is responsible for executing all the different functions needed to adapt the UI to the capabilities and constraints of the device, application and/or user preferences. Maes at 36:49-37:3: The conversational dialog manager 819 comprises functions for managing the dialog (conversational dialog comprising speech and other multi-modal I/O such as GUI keyboard, pointer, mouse, as well as video input, etc.) and arbitration (dialog manager arbitrator or DMA) across all registered applications. In particular, the conversational dialog manager 819 determines what information the user has, which inputs the user presents, and which application(s) should

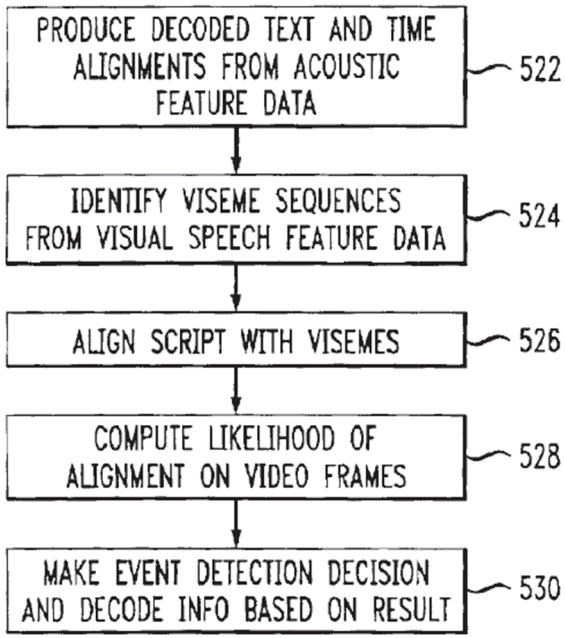
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	handle the user inputs. The DMA processes abstracted I/O events (abstracted by the I/O manager) using the context/history to understand the user intent. When an abstract event occurs, the DMA determines the target of the event and, if needed, seeks confirmation, disambiguation, correction, more details, etc., until the intent is unambiguous and fully determined. The DMA then launches the action associated to the user's query. The DMA function handles multi-modal I/O events to: (1) determine the target application or dialog (or portion of it); and (2) use past history and context to: (a) understand the intent of the user; (b) follow up with a dialog to disambiguate, complete, correct or confirm the understanding; (c) or, dispatch a task resulting from fall understanding of the intent of the user. Maes at 41:48-42:7: Apparatus 900 can further include a post processor 938 which is coupled to the data warehouse 912 and which is configured to transcribe user utterances and to perform keyword spotting thereon. Although shown as a separate item in FIG. 9A, the post processor can be a part of the processing module 910 or of any of the sub-components thereof. For example, it can be implemented as part of the speech recognizor 926. Post processor 938 can be implemented as part of the processor of an IVR, as an application specific circuit, or on a general purpose computer with suitable software modules. Post processor 938 can employ speech recognizor 926. Post processor 938 can also include a semantic module (not shown) to interpret meaning of phrases. The semantic module could be used by speech recognizor 926 to indicate that some decoding candidates in a list are meaningless and should be discarded/replaced with meaningful candidates. The acoustic front end 908 can typically be an eight dimensions plus energy front end as known in the art. However, it should be understood that 13, 24, or any other number of dimensions could be used. MEL cepstra can be computed, for example, over 25 ms frames with a 10 ms overlap, along with the delta and delta delta parameters, that is, the first and second finite derivatives. Such acoustic features can be supplied to the speaker clusterer and classifier 920, speech recognizor 926 and accent identifier 932, as shown in FIG. 9A. Maes at 42:20-23:

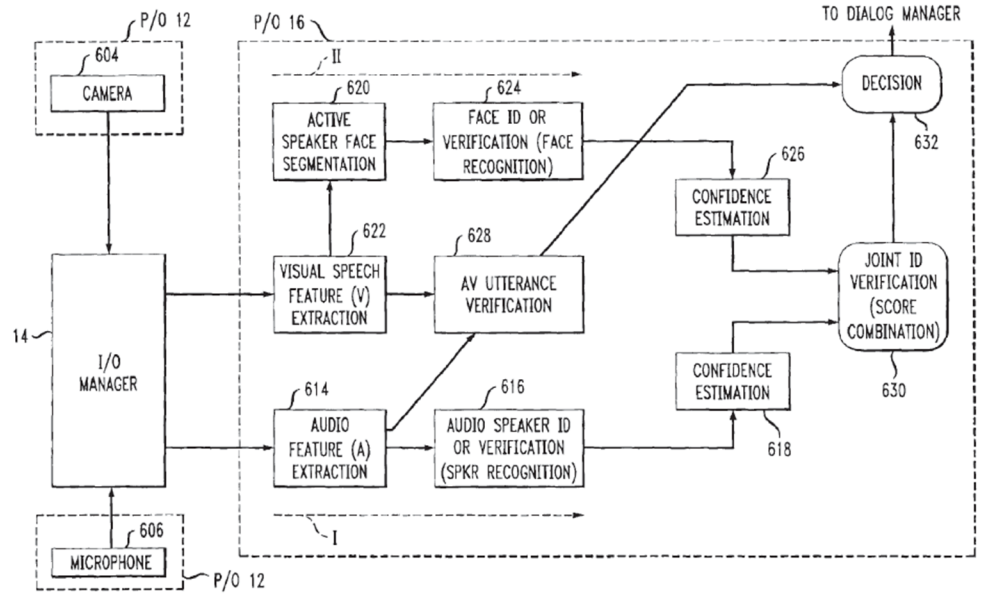
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	User queries can be transcribed by an IVR or otherwise. Speech features can first be processed by a text-independent speaker classification system, for example, in speaker clusterer and classifier 920. Maes at 42:57-43:2: The speech recognizor 926 can transcribe the queries from the user. It can be a speaker-independent or class-dependent large vocabulary continuous speech recognition, or system could be something as simple as a keyword spotter to detect insults (for example) and the like. Such systems are well known in the art. The output can be full sentences, but finer granularity can also be attained; for example, time alignment of the recognized words. The time stamped transcriptions can also be considered as part of the processed features, and will be discussed further below with respect to methods in accordance with the present invention. Thus, conversation from every stage of a transaction can be transcribed and stored. To the extent it is found that Maes does not expressly disclose this limitation, the limitation is inherent. And to the extent it is found that Maes does not anticipate this claim, Maes renders it obvious, standing alone, in combination with the knowledge of a person of ordinary skill in the art, and/or in combination with one or more other prior art references identified in the Cover Pleading or herein, for the reasons identified herein or in the Cover Pleading. A person of ordinary skill in the art would have been motivated to combine Maes with any of the other prior art references identified in the Cover Pleading or herein with a reasonable expectation of success because each of these references contains familiar elements within the similar fields of endeavor to address similar known problems, according to known methods which would yield predictable results.
[d] identifying, from among the plurality of context entries, one or more context entries that correspond to the one or more words, wherein the context information includes	Maes discloses "identifying, from among the plurality of context entries, one or more context entries that correspond to the one or more words, wherein the context information includes the one or more context entries[.]" *E.g.*, Maes at Fig. 2:

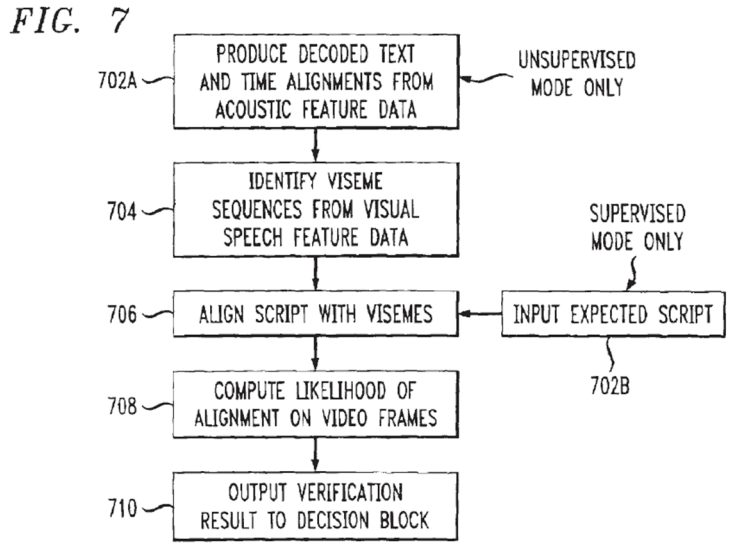
the one or more context entries, wherein identifying the one or more context entries comprises:	

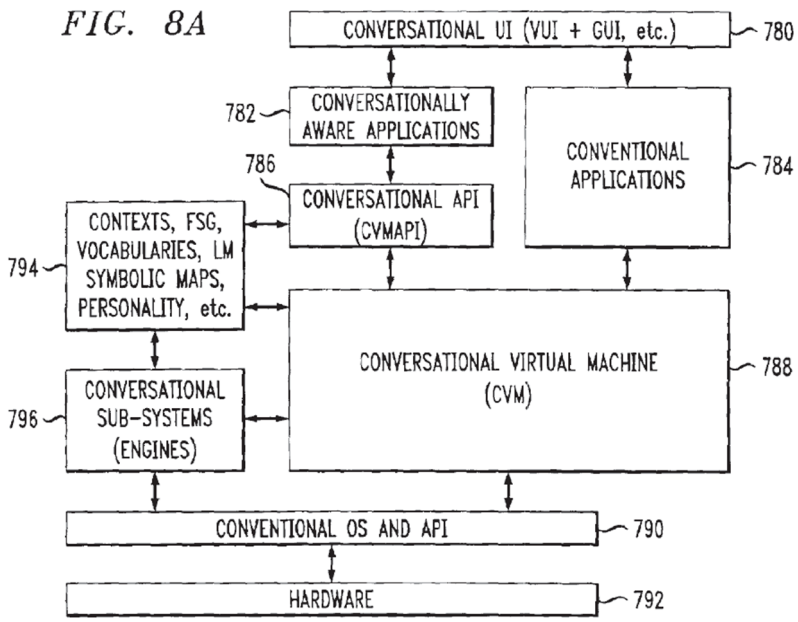
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	Maes at Fig. 3:  Maes at Fig. 4:

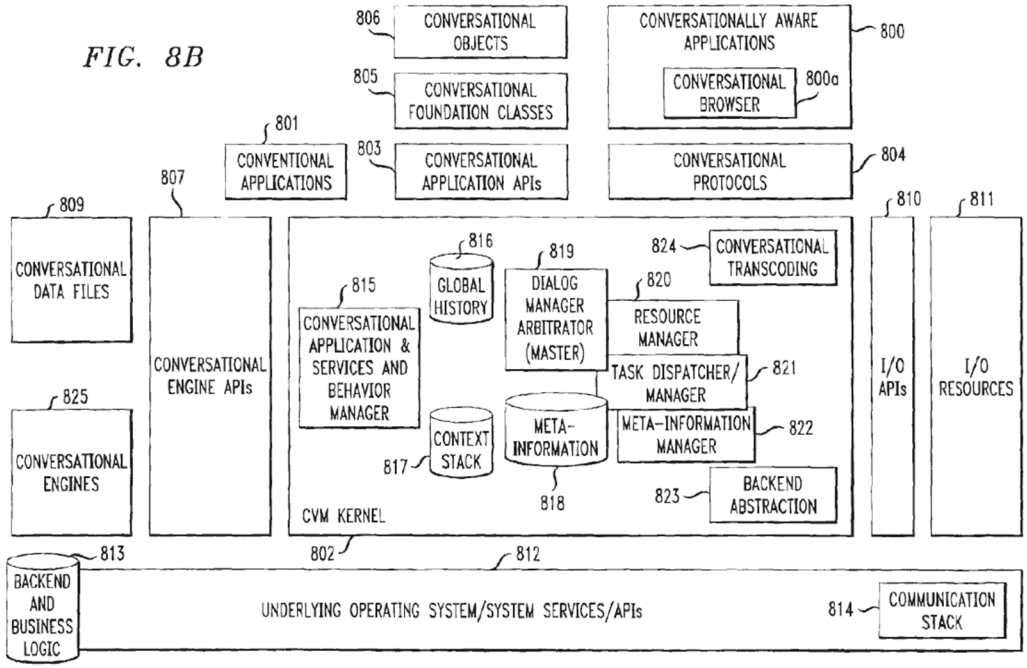
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at 5D:

Claim 7 of the '957 Patent	Exemplary Citations to Maes
	FIG. 5D  Maes at Fig. 6:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	FIG. 6  Maes at Fig. 7:

Claim 7 of the '957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 8A:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig 8B:

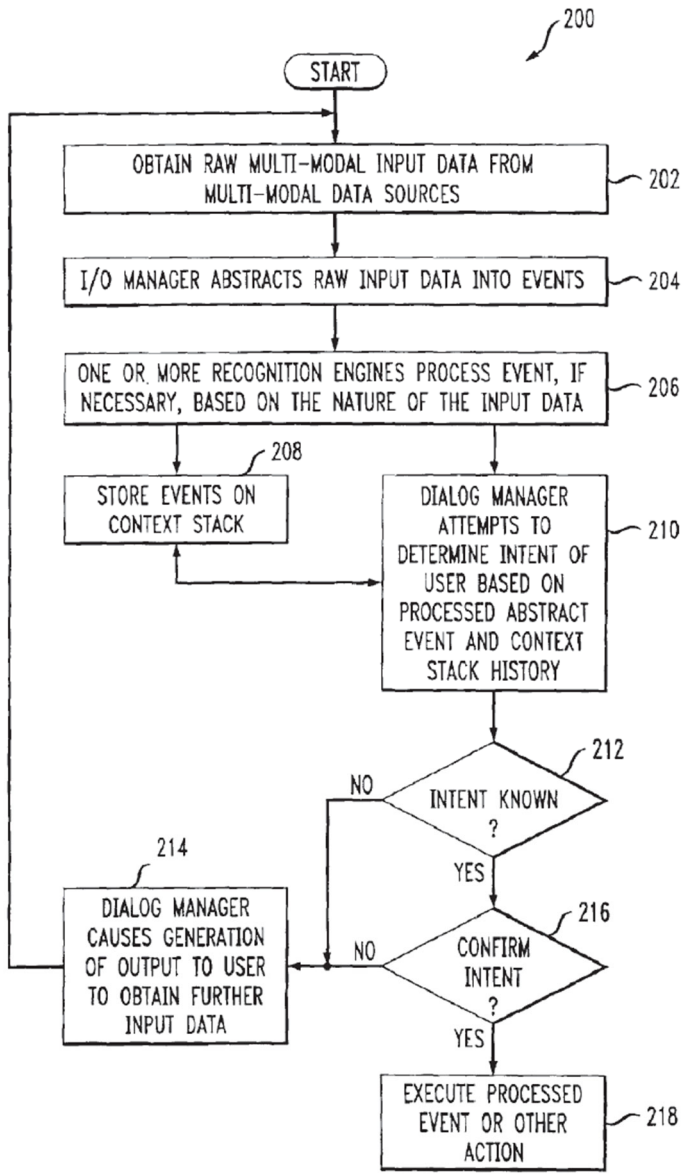
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	 Maes at 12:29-48: After the acoustic feature vectors, denoted in FIG. 4. by the letter A, are extracted, the probability module labels the extracted vectors with one or more previously stored phonemes which, as is known in the art, are sub-phonetic or acoustic units of speech. The module may also work with lefemes, which are portions of phones in a given context. Each phoneme associated with one or more feature vectors has a probability associated therewith indicating the likelihood that it was that particular acoustic unit that was spoken. Thus, the probability module yields likelihood scores for each considered phoneme in the form of the probability that, given a particular phoneme or acoustic unit (au), the acoustic unit represents the uttered speech characterized by one or more acoustic feature vectors A or, in other words, P(A|acoustic unit). It is to be appreciated that the

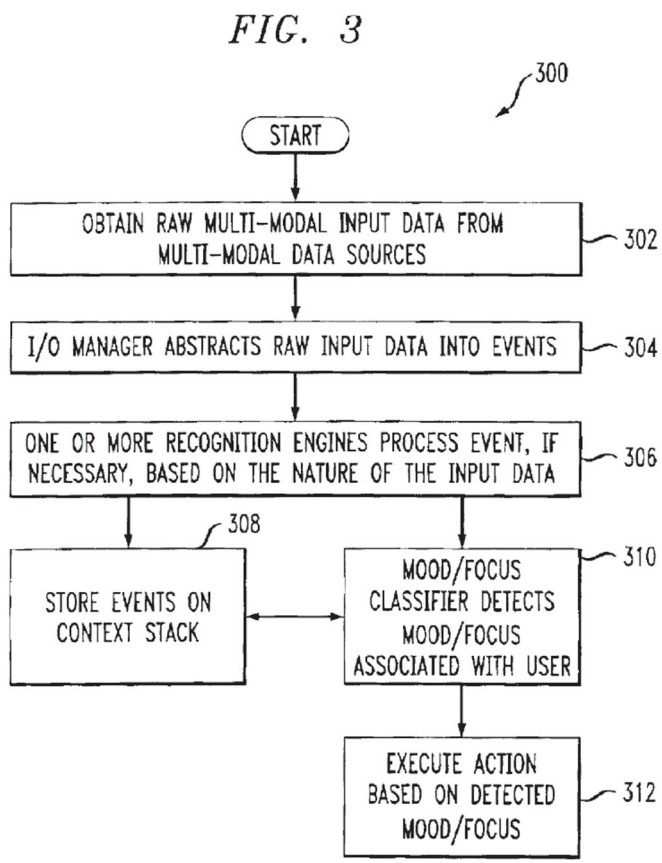
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	processing performed in blocks 414 and 416 may be accomplished via any conventional acoustic information recognition system capable of extracting and labeling acoustic feature vectors, e.g., Lawrence Rabiner, Biing-Hwang Juang, “Fundamentals of Speech Recognition,” Prentice Hall, 1993. Maes at 20:11-20: Then, a search is performed in search module 434 with language models (LM) based on the weighted probabilities received from module 430. That is, the acoustic units identified as having the highest probabilities of representing what was uttered in the arbitrary content video are put together to form words. The words are output by the search engine 434 as the decoded system output. A conventional search engine may be employed. This output is provided to the dialog manager 18 of FIG. 1 for use in disambiguating the user's intent, as described above. Maes at 36:8-48: The core CVM kernel layer 802 comprises programming layers such as a conversational application and behavior/service manager layer 815, a conversational dialog manager (arbitrator) layer 819, a conversational resource manager layer 820, a task/dispatcher manager 821 and a meta-information manager 822, which provide the core functions of the CVM layer 802. It is to be understood that these components may be implemented as part of the dialog manager 18 of FIG. 1. The conversational application and behavior/service manager layer 815 comprises functions for managing the conventional and conversationally aware applications 800 and 801. Such management functions include, for example, keeping track of which applications are registered (both local and network-distributed), what are the dialog interfaces (if any) of the applications, and what is the state of each application. In addition, the conversational application and services/behavior manager 815 initiates all the tasks associated with any specific service or behavior provided by the CVM system. The conversational services and behaviors are all the behaviors and features of a conversational UI that the user may expect to find in the applications and interactions, as well as the features that an application developer may expect to be able to access via APIs (without having to implement with the development of the application). Examples

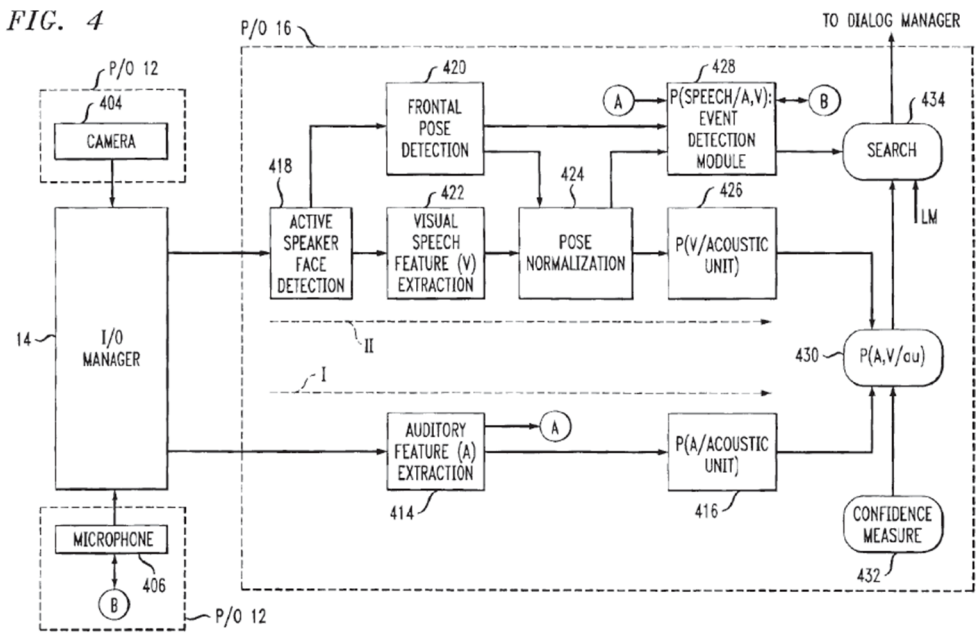
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	of the conversational services and behavior provided by the CVM kernel 802 include, but are not limited to, conversational categorization and meta-information, conversational object, resource and file management, conversational search, conversational selection, conversational customization, conversational security, conversational help, conversational prioritization, conversational resource management, output formatting and presentation, summarization, conversational delayed actions/agents/memorization, conversational logic, and coordinated interfaces and devices. Such services are provided through API calls via the conversational application API Layer 803. The conversational application and behavior/services manager 815 is responsible for executing all the different functions needed to adapt the UI to the capabilities and constraints of the device, application and/or user preferences. Maes at 36:49-37:3: The conversational dialog manager 819 comprises functions for managing the dialog (conversational dialog comprising speech and other multi-modal I/O such as GUI keyboard, pointer, mouse, as well as video input, etc.) and arbitration (dialog manager arbitrator or DMA) across all registered applications. In particular, the conversational dialog manager 819 determines what information the user has, which inputs the user presents, and which application(s) should handle the user inputs. The DMA processes abstracted I/O events (abstracted by the I/O manager) using the context/history to understand the user intent. When an abstract event occurs, the DMA determines the target of the event and, if needed, seeks confirmation, disambiguation, correction, more details, etc., until the intent is unambiguous and fully determined. The DMA then launches the action associated to the user's query. The DMA function handles multi-modal I/O events to: (1) determine the target application or dialog (or portion of it); and (2) use past history and context to: (a) understand the intent of the user; (b) follow up with a dialog to disambiguate, complete, correct or confirm the understanding; (c) or, dispatch a task resulting from fall understanding of the intent of the user. Maes at 38:3-17:

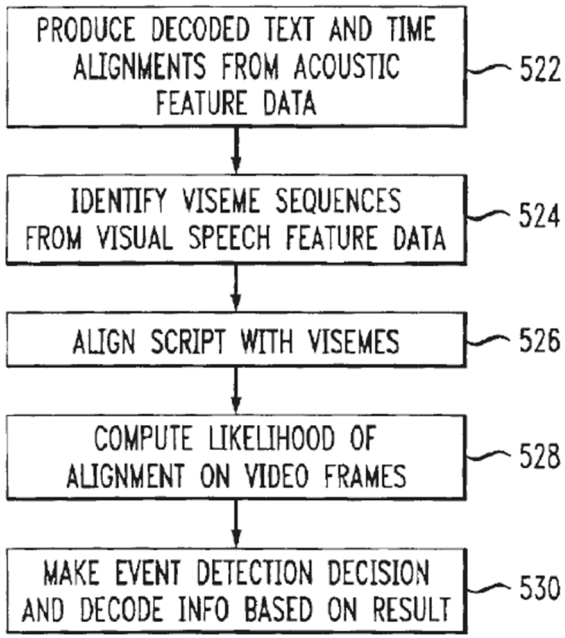
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	The CVM kernel layer 802 further comprises a backend abstraction layer 823 which allows access to backend business logic 813 via the dialog manager 819 (rather than bypassing the dialog manager 819). This allows such accesses to be added to the context stack 817 and global history 816. For instance, the backend abstraction layer 823 can translate input and output to and from the dialog manager 819 to database queries. This layer 823 will convert standardized attribute value n-tuples into database queries and translate the result of such queries into tables or sets of attribute value n-tuples back to the dialog manager 819. In addition, a conversational transcoding layer 824 is provided to adapt the behavior, UI and dialog presented to the user based on the I/O and engine capabilities of the device which executes the CVM system. To the extent it is found that Maes does not expressly disclose this limitation, the limitation is inherent. And to the extent it is found that Maes does not anticipate this claim, Maes renders it obvious, standing alone, in combination with the knowledge of a person of ordinary skill in the art, and/or in combination with one or more other prior art references identified in the Cover Pleading or herein, for the reasons identified herein or in the Cover Pleading. A person of ordinary skill in the art would have been motivated to combine Maes with any of the other prior art references identified in the Cover Pleading or herein with a reasonable expectation of success because each of these references contains familiar elements within the similar fields of endeavor to address similar known problems, according to known methods which would yield predictable results.
[e] comparing the plurality of context entries to the one or more words;	Maes discloses "comparing the plurality of context entries to the one or more words[.]" *E.g.*, Maes at Fig. 2:

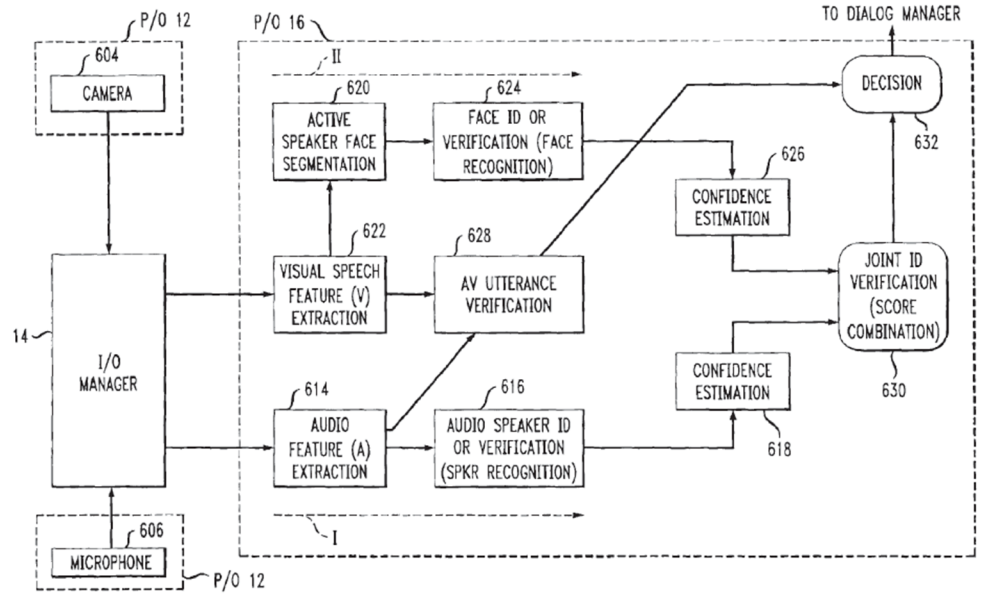
FIG. 2

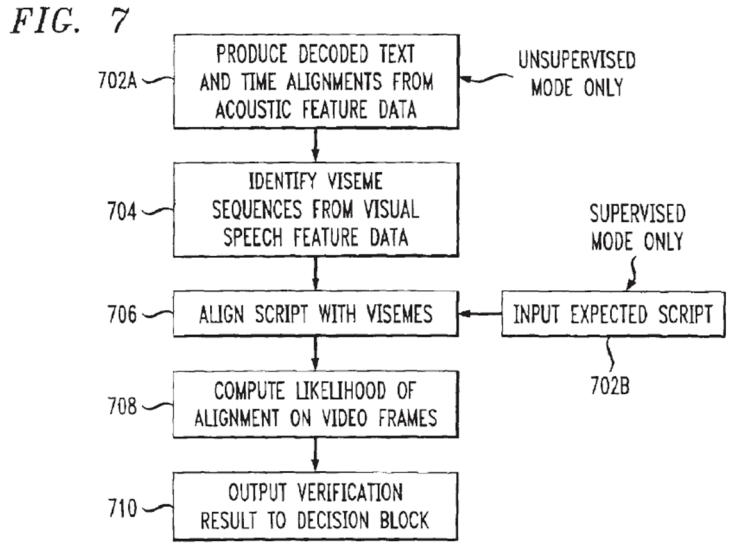

200
START
OBTAIN RAW MULTI-MODAL INPUT DATA FROM MULTI-MODAL DATA SOURCES
202
I/O MANAGER ABSTRACTS RAW INPUT DATA INTO EVENTS
204
ONE OR MORE RECOGNITION ENGINES PROCESS EVENT, IF NECESSARY, BASED ON THE NATURE OF THE INPUT DATA
206
208
STORE EVENTS ON CONTEXT STACK
DIALOG MANAGER ATTEMPTS TO DETERMINE INTENT OF USER BASED ON PROCESSED ABSTRACT EVENT AND CONTEXT STACK HISTORY
210
212
NO
INTENT KNOWN ?
YES
214
DIALOG MANAGER CAUSES GENERATION OF OUTPUT TO USER TO OBTAIN FURTHER INPUT DATA
216
NO
CONFIRM INTENT ?
YES
EXECUTE PROCESSED EVENT OR OTHER ACTION
218

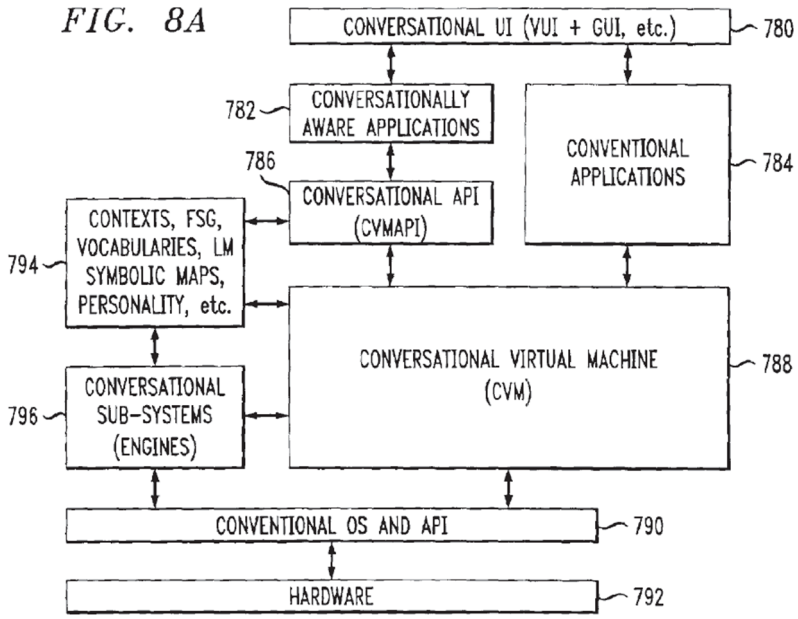
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	Maes at Fig. 3:  Maes at Fig. 4:

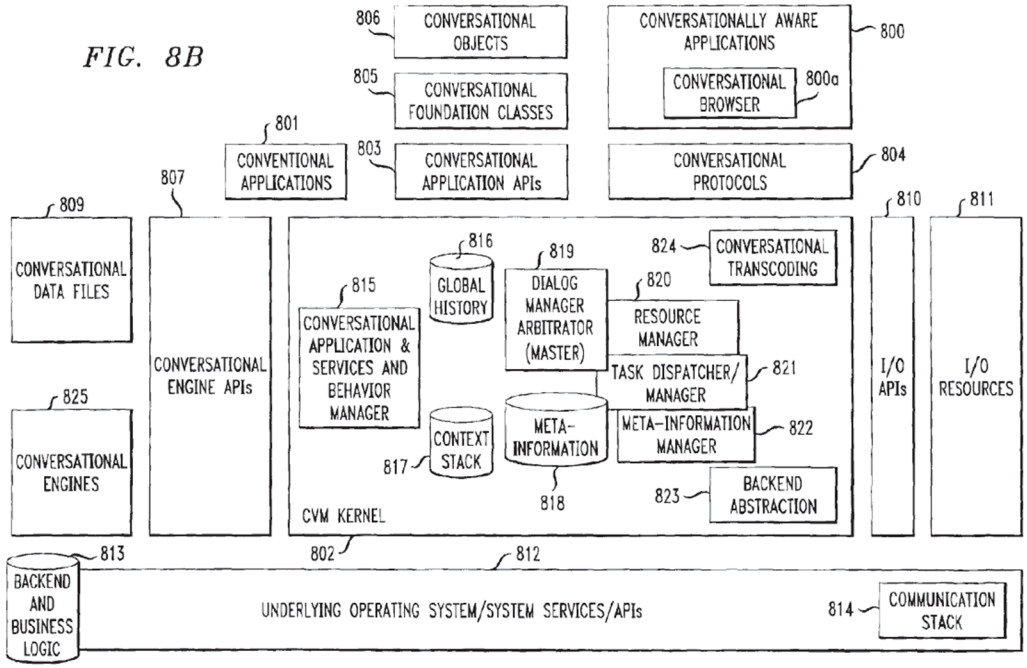
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at 5D:

Claim 7 of the '957 Patent	Exemplary Citations to Maes
	FIG. 5D  Maes at Fig. 6:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	FIG. 6  Maes at Fig. 7:

Claim 7 of the '957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 8A:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig 8B:

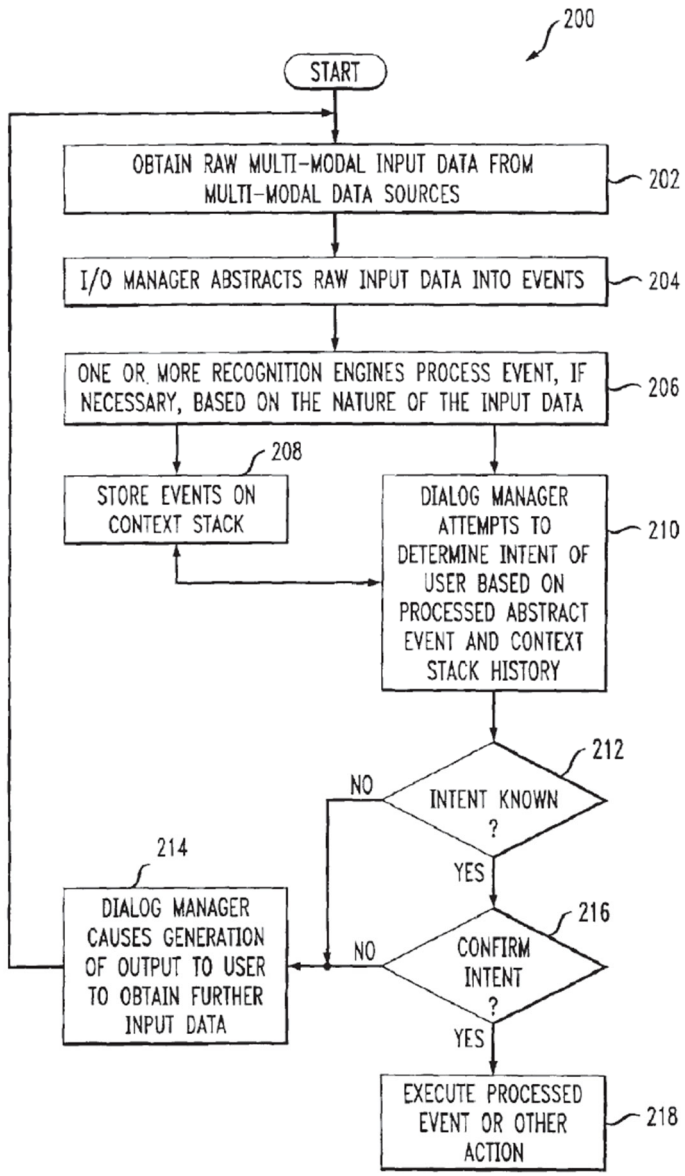
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	 Maes at 12:29-48: After the acoustic feature vectors, denoted in FIG. 4. by the letter A, are extracted, the probability module labels the extracted vectors with one or more previously stored phonemes which, as is known in the art, are sub-phonetic or acoustic units of speech. The module may also work with lefemes, which are portions of phones in a given context. Each phoneme associated with one or more feature vectors has a probability associated therewith indicating the likelihood that it was that particular acoustic unit that was spoken. Thus, the probability module yields likelihood scores for each considered phoneme in the form of the probability that, given a particular phoneme or acoustic unit (au), the acoustic unit represents the uttered speech characterized by one or more acoustic feature vectors A or, in other words, P(A|acoustic unit). It is to be appreciated that the

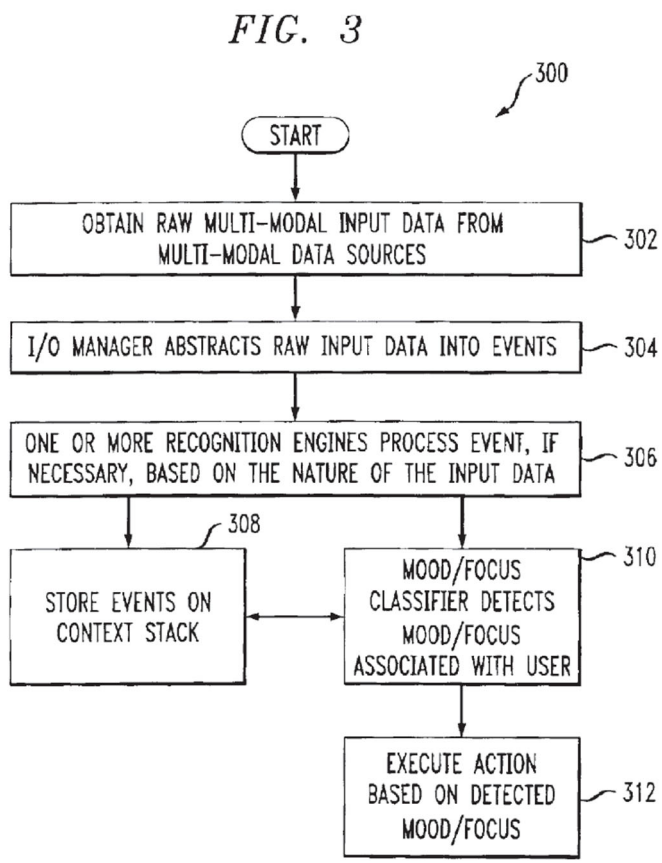
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	processing performed in blocks 414 and 416 may be accomplished via any conventional acoustic information recognition system capable of extracting and labeling acoustic feature vectors, e.g., Lawrence Rabiner, Biing-Hwang Juang, “Fundamentals of Speech Recognition,” Prentice Hall, 1993. Maes at 20:11-20: Then, a search is performed in search module 434 with language models (LM) based on the weighted probabilities received from module 430. That is, the acoustic units identified as having the highest probabilities of representing what was uttered in the arbitrary content video are put together to form words. The words are output by the search engine 434 as the decoded system output. A conventional search engine may be employed. This output is provided to the dialog manager 18 of FIG. 1 for use in disambiguating the user's intent, as described above. Maes at 36:8-48: The core CVM kernel layer 802 comprises programming layers such as a conversational application and behavior/service manager layer 815, a conversational dialog manager (arbitrator) layer 819, a conversational resource manager layer 820, a task/dispatcher manager 821 and a meta-information manager 822, which provide the core functions of the CVM layer 802. It is to be understood that these components may be implemented as part of the dialog manager 18 of FIG. 1. The conversational application and behavior/service manager layer 815 comprises functions for managing the conventional and conversationally aware applications 800 and 801. Such management functions include, for example, keeping track of which applications are registered (both local and network-distributed), what are the dialog interfaces (if any) of the applications, and what is the state of each application. In addition, the conversational application and services/behavior manager 815 initiates all the tasks associated with any specific service or behavior provided by the CVM system. The conversational services and behaviors are all the behaviors and features of a conversational UI that the user may expect to find in the applications and interactions, as well as the features that an application developer may expect to be able to access via APIs (without having to implement with the development of the application). Examples

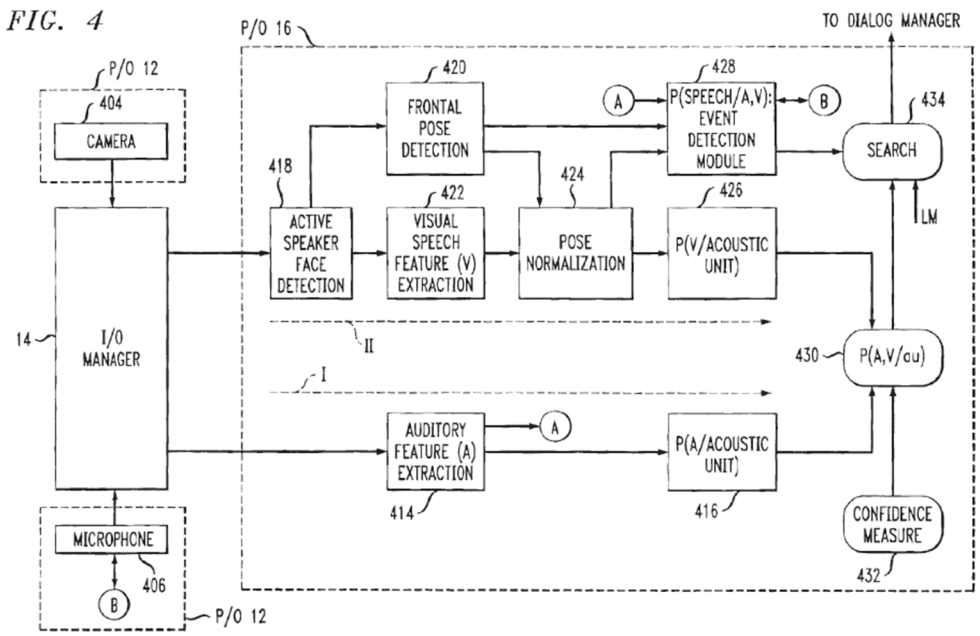
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	of the conversational services and behavior provided by the CVM kernel 802 include, but are not limited to, conversational categorization and meta-information, conversational object, resource and file management, conversational search, conversational selection, conversational customization, conversational security, conversational help, conversational prioritization, conversational resource management, output formatting and presentation, summarization, conversational delayed actions/agents/memorization, conversational logic, and coordinated interfaces and devices. Such services are provided through API calls via the conversational application API Layer 803. The conversational application and behavior/services manager 815 is responsible for executing all the different functions needed to adapt the UI to the capabilities and constraints of the device, application and/or user preferences. Maes at 36:49-37:3: The conversational dialog manager 819 comprises functions for managing the dialog (conversational dialog comprising speech and other multi-modal I/O such as GUI keyboard, pointer, mouse, as well as video input, etc.) and arbitration (dialog manager arbitrator or DMA) across all registered applications. In particular, the conversational dialog manager 819 determines what information the user has, which inputs the user presents, and which application(s) should handle the user inputs. The DMA processes abstracted I/O events (abstracted by the I/O manager) using the context/history to understand the user intent. When an abstract event occurs, the DMA determines the target of the event and, if needed, seeks confirmation, disambiguation, correction, more details, etc., until the intent is unambiguous and fully determined. The DMA then launches the action associated to the user's query. The DMA function handles multi-modal I/O events to: (1) determine the target application or dialog (or portion of it); and (2) use past history and context to: (a) understand the intent of the user; (b) follow up with a dialog to disambiguate, complete, correct or confirm the understanding; (c) or, dispatch a task resulting from fall understanding of the intent of the user. Maes at 38:3-17:

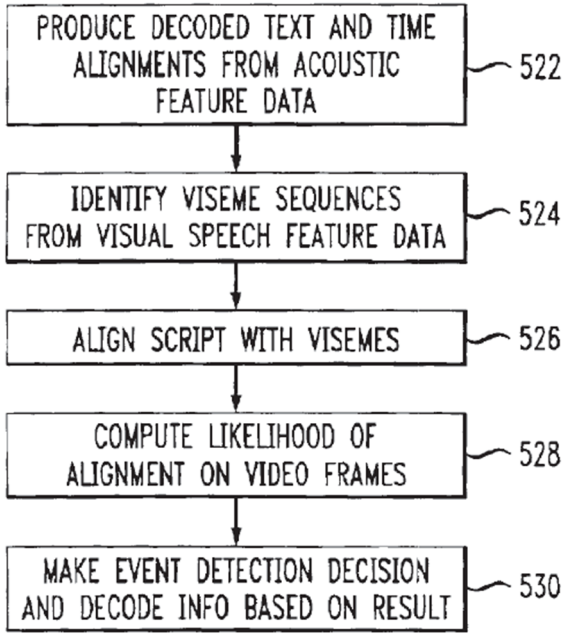
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	The CVM kernel layer 802 further comprises a backend abstraction layer 823 which allows access to backend business logic 813 via the dialog manager 819 (rather than bypassing the dialog manager 819). This allows such accesses to be added to the context stack 817 and global history 816. For instance, the backend abstraction layer 823 can translate input and output to and from the dialog manager 819 to database queries. This layer 823 will convert standardized attribute value n-tuples into database queries and translate the result of such queries into tables or sets of attribute value n-tuples back to the dialog manager 819. In addition, a conversational transcoding layer 824 is provided to adapt the behavior, UI and dialog presented to the user based on the I/O and engine capabilities of the device which executes the CVM system. To the extent it is found that Maes does not expressly disclose this limitation, the limitation is inherent. And to the extent it is found that Maes does not anticipate this claim, Maes renders it obvious, standing alone, in combination with the knowledge of a person of ordinary skill in the art, and/or in combination with one or more other prior art references identified in the Cover Pleading or herein, for the reasons identified herein or in the Cover Pleading. A person of ordinary skill in the art would have been motivated to combine Maes with any of the other prior art references identified in the Cover Pleading or herein with a reasonable expectation of success because each of these references contains familiar elements within the similar fields of endeavor to address similar known problems, according to known methods which would yield predictable results.
[f] generating, based on the comparison, one or more rank scores for individual context entries of the plurality of context entries; and	Maes discloses "generating, based on the comparison, one or more rank scores for individual context entries of the plurality of context entries[.]" *E.g.*, Maes at Fig. 2:

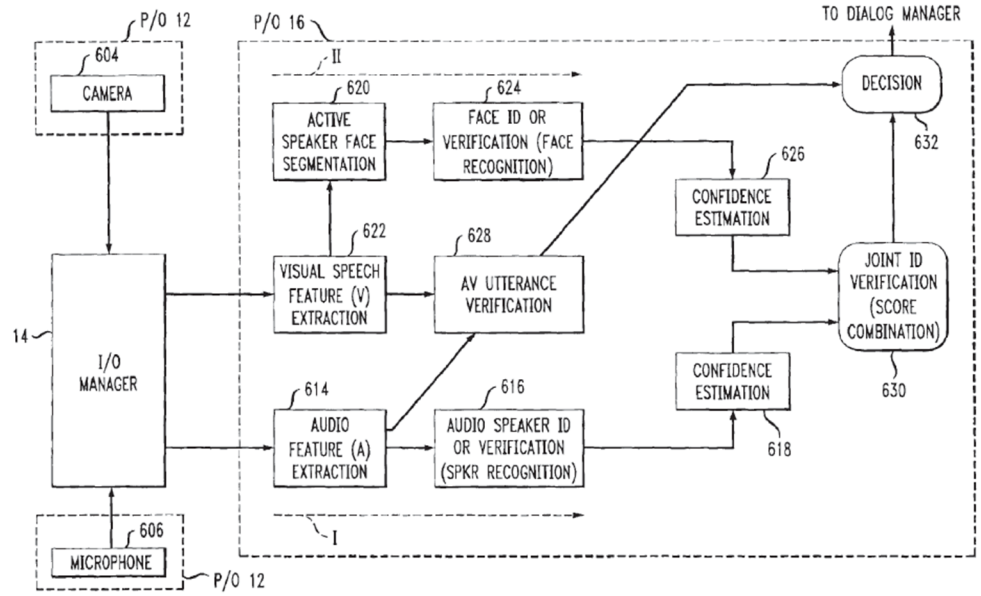
FIG. 2

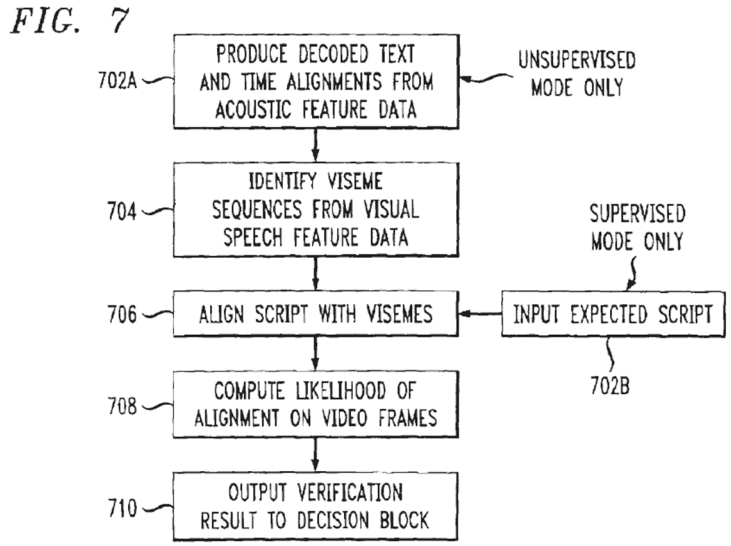

200
START
OBTAIN RAW MULTI-MODAL INPUT DATA FROM MULTI-MODAL DATA SOURCES
202
I/O MANAGER ABSTRACTS RAW INPUT DATA INTO EVENTS
204
ONE OR MORE RECOGNITION ENGINES PROCESS EVENT, IF NECESSARY, BASED ON THE NATURE OF THE INPUT DATA
206
208
STORE EVENTS ON CONTEXT STACK
DIALOG MANAGER ATTEMPTS TO DETERMINE INTENT OF USER BASED ON PROCESSED ABSTRACT EVENT AND CONTEXT STACK HISTORY
210
212
NO
INTENT KNOWN ?
YES
214
DIALOG MANAGER CAUSES GENERATION OF OUTPUT TO USER TO OBTAIN FURTHER INPUT DATA
216
NO
CONFIRM INTENT ?
YES
EXECUTE PROCESSED EVENT OR OTHER ACTION
218

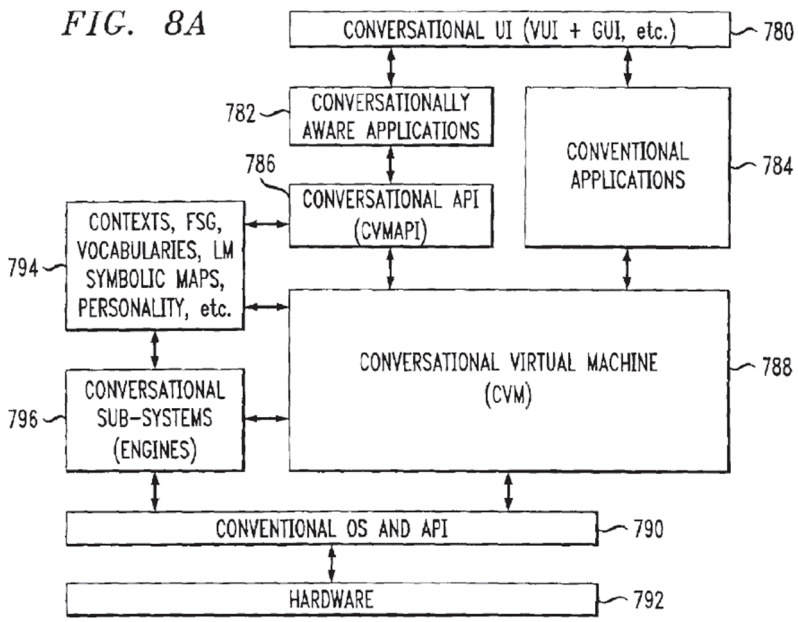
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	Maes at Fig. 3:  Maes at Fig. 4:

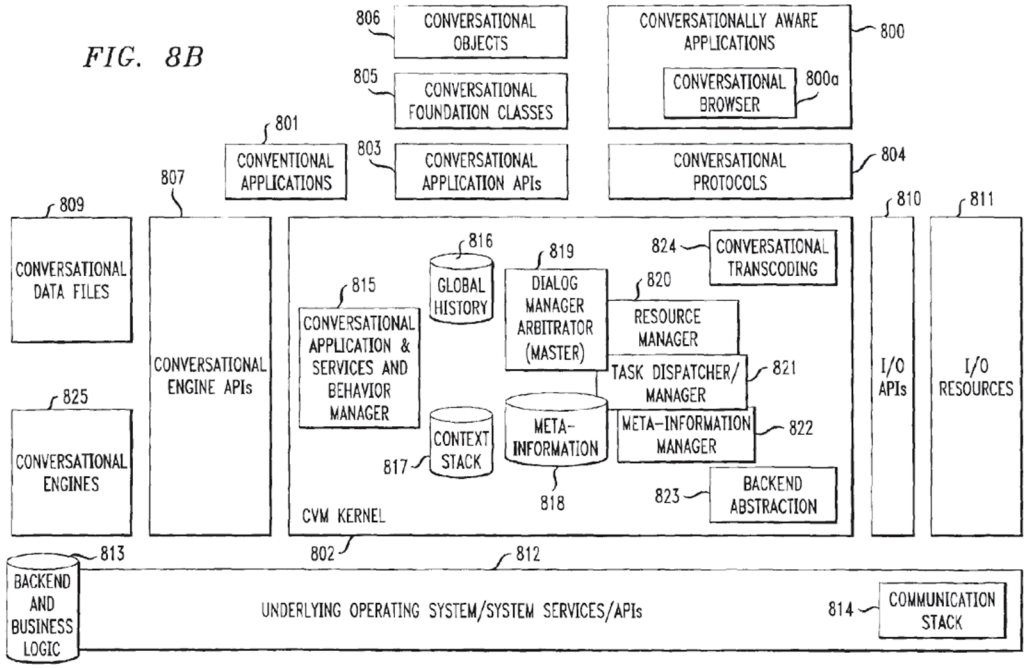
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at 5D:

Claim 7 of the '957 Patent	Exemplary Citations to Maes
	FIG. 5D  Maes at Fig. 6:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	FIG. 6  Maes at Fig. 7:

Claim 7 of the '957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 8A:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig 8B:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at 7:60-8:65: Thus, the dialog manager must first determine the user's intent based on the current event and, if available, the historical information (e.g., past events) stored in the context stack. For instance, returning to the vehicle example, the user may say “turn it on,” while pointing at the vehicle radio. The dialog manager would therefore receive the results of the recognized events associated with the spoken utterance “turn it on” and the gesture of pointing to the radio. Based on these events, the dialog manager does a search of the existing applications, transactions or “dialogs,” or portions thereof, with which such an utterance and gesture could be associated. Accordingly, as shown in FIG. 1, the dialog manager 18 determines the appropriate CML-authored application 24. The application may be stored on the system 10 or accessed (e.g., downloaded) from some remote

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	location. If the dialog manager determines with some predetermined degree of confidence that the application it selects is the one which will effectuate the users desire, the dialog manager executes the next step of the multi-modal dialog (e.g., prompt or display for missing, ambiguous or confusing information, asks for confirmation or launches the execution of an action associated to a fully understood multi-modal request from the user) of that application based on the multi-modal input. That is, the dialog manager selects the appropriate device (e.g., radio) activation routine and instructs the I/O manager to output a command to activate the radio. The predetermined degree of confidence may be that at least two input parameters or variables of the application are satisfied or provided by the received events. Of course, depending on the application, other levels of confidence and algorithms may be established as, for example, described in K. A. Papineni, S. Roukos, R. T. Ward, “Free-flow dialog management using forms,” Proc. Eurospeech, Budapest, 1999; and K. Davies et al., “The IBM conversational telephony system for financial applications,” Proc. Eurospeech, Budapest, 1999, the disclosures of which are incorporated by reference herein. Consider the case where the user first says “turn it on,” and then a few seconds later points to the radio. The dialog manager would first try to determine user intent based solely on the “turn it on” command. However, since there are likely other devices in the vehicle that could be turned on, the system would likely not be able to determine with a sufficient degree of confidence what the user was referring to. However, this recognized spoken utterance event is stored on the context stack. Then, when the recognized gesture event (e.g., pointing to the radio) is received, the dialog manager takes this event and the previous spoken utterance event stored on the context stack and makes a determination that the user intended to have the radio turned on. Consider the case where the user says “turn it on,” but makes no gesture and provides no other utterance. In this case, assume that the dialog manager does not have enough input to determine the user intent (step 212 in FIG. 2) and thus implement the command. The dialog manager, in step 214, then causes the generation of an output to the user requesting further input data so that the user's intent can be disambiguated. This may be accomplished by the dialog manager instructing the I/O manager to have the I/O subsystem output a request for clarification. In one embodiment, the I/O subsystem 12 may comprise a text-to-speech (TTS) engine and one or more output speakers. The dialog manager then generates a predetermined question such as “what device do

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	you want to have turned on?” which the TTS engine converts to a synthesized utterance that is audibly output by the speaker to the user. The user, hearing the query, could then point to the radio or say “the radio” thereby providing the dialog manager with the additional input data to disambiguate his request. That is, with reference to FIG. 2, the system 10 obtains the raw input data, again in step 202, and the process 200 iterates based on the new data. Such iteration can continue as long as necessary for the dialog manager to determine the user's intent. Maes at 12:29-48: After the acoustic feature vectors, denoted in FIG. 4. by the letter A, are extracted, the probability module labels the extracted vectors with one or more previously stored phonemes which, as is known in the art, are sub-phonetic or acoustic units of speech. The module may also work with lefemes, which are portions of phones in a given context. Each phoneme associated with one or more feature vectors has a probability associated therewith indicating the likelihood that it was that particular acoustic unit that was spoken. Thus, the probability module yields likelihood scores for each considered phoneme in the form of the probability that, given a particular phoneme or acoustic unit (au), the acoustic unit represents the uttered speech characterized by one or more acoustic feature vectors A or, in other words, P(A|acoustic unit). It is to be appreciated that the processing performed in blocks 414 and 416 may be accomplished via any conventional acoustic information recognition system capable of extracting and labeling acoustic feature vectors, e.g., Lawrence Rabiner, Biing-Hwang Juang, “Fundamentals of Speech Recognition,” Prentice Hall, 1993. Maes at 18:64-20:10: Next, the probabilities generated by modules 416 and 426 are jointly used by A, V probability module 430. In module 430, the respective probabilities from modules 416 and 426 are combined based on a confidence measure 432. Confidence estimation refers to a likelihood or other confidence measure being determined with regard to the recognized input. Recently, efforts have been initiated to develop appropriate confidence measures for recognized speech. In LVCSR Hub5 Workshop, Apr. 29-May 1, 1996, MITAGS, MD, organized by NIST and DARPA, different

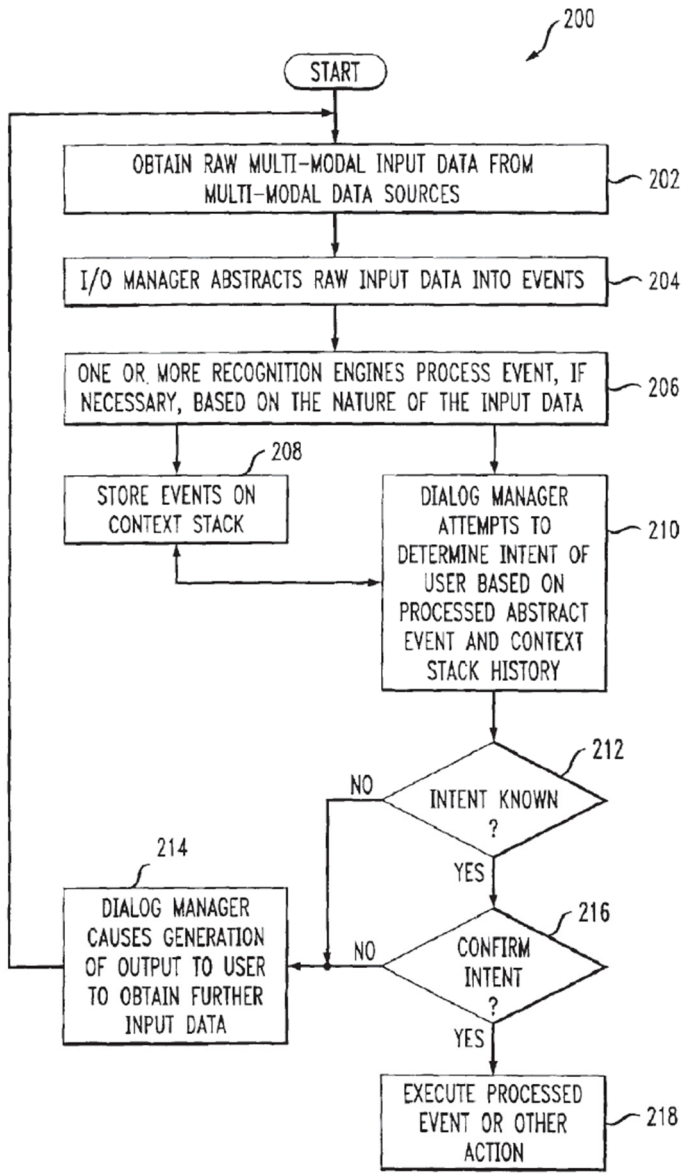
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	approaches are proposed to attach to each word a confidence level. A first method uses decision trees trained on word-dependent features (amount of training utterances, minimum and average triphone occurrences, occurrence in language model training, number of phonemes/lefemes, duration, acoustic score (fast match and detailed match), speech or non-speech), sentence-dependent features (signal-to-noise ratio, estimates of speaking rates: number of words or of lefemes or of vowels per second, sentence likelihood provided by the language model, trigram occurrence in the language model), word in a context features (trigram occurrence in language model) as well as speaker profile features (accent, dialect, gender, age, speaking rate, identity, audio quality, SNR, etc . . .). A probability of error is computed on the training data for each of the leaves of the tree. Algorithms to build such trees are disclosed, for example, in Breiman et al., “Classification and regression trees,” Chapman & Hall,1993. At recognition, all or some of these features are measured during recognition and for each word the decision tree is walked to a leave which provides a confidence level. In C. Neti, S. Roukos and E. Eide “Word based confidence measures as a guide for stack search in speech recognition,” ICASSP97, Munich, Germany, April, 1997, is described a method relying entirely on scores returned by IBM stack decoder (using log-likelihood—actually the average incremental log-likelihood, detailed match, fast match). In the LVCSR proceeding, another method to estimate the confidence level is done using predictors via linear regression. The predictor used are: the word duration, the language model score, the average acoustic score (best score) per frame and the fraction of the N-Best list with the same word as top choice. The present embodiment preferably offers a combination of these two approaches (confidence level measured via decision trees and via linear predictors) to systematically extract the confidence level in any translation process, not limited to speech recognition. Another method to detect incorrectly recognized words is disclosed in U.S. Pat. No. 5,937,383 entitled “Apparatus and Methods for Speech Recognition Including Individual or Speaker Class Dependent Decoding History Caches for Fast Word Acceptance or Rejection,” the disclosure of which is incorporated herein by reference.

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Thus, based on the confidence measure, the probability module 430 decides which probability, i.e., the probability from the visual information path or the probability from the audio information path, to rely on more. This determination may be represented in the following manner: $w_1 v_P + w_2 a_P$. It is to be understood that vP represents a probability associated with the visual information, aP represents a probability associated with the corresponding audio information, and w1 and w2 represent respective weights. Thus, based on the confidence measure 432, the module 430 assigns appropriate weights to the probabilities. For instance, if the surrounding environmental noise level is particularly high, i.e., resulting in a lower acoustic confidence measure, there is more of a chance that the probabilities generated by the acoustic decoding path contain errors. Thus, the module 430 assigns a lower weight for w2 than for w1 placing more reliance on the decoded information from the visual path. However, if the noise level is low and thus the acoustic confidence measure is relatively higher, the module may set w2 higher than w1. Alternatively, a visual confidence measure may be used. It is to be appreciated that the first joint use of the visual information and audio information in module 430 is referred to as decision or score fusion. An alternative embodiment implements feature fusion as described in the above-referenced U.S. patent application identified as Ser. No. 09/369,707. Maes at 36:8-48: The core CVM kernel layer 802 comprises programming layers such as a conversational application and behavior/service manager layer 815, a conversational dialog manager (arbitrator) layer 819, a conversational resource manager layer 820, a task/dispatcher manager 821 and a meta-information manager 822, which provide the core functions of the CVM layer 802. It is to be understood that these components may be implemented as part of the dialog manager 18 of FIG. 1. The conversational application and behavior/service manager layer 815 comprises functions for managing the conventional and conversationally aware applications 800 and 801. Such management functions include, for example, keeping track of which applications are registered

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	(both local and network-distributed), what are the dialog interfaces (if any) of the applications, and what is the state of each application. In addition, the conversational application and services/behavior manager 815 initiates all the tasks associated with any specific service or behavior provided by the CVM system. The conversational services and behaviors are all the behaviors and features of a conversational UI that the user may expect to find in the applications and interactions, as well as the features that an application developer may expect to be able to access via APIs (without having to implement with the development of the application). Examples of the conversational services and behavior provided by the CVM kernel 802 include, but are not limited to, conversational categorization and meta-information, conversational object, resource and file management, conversational search, conversational selection, conversational customization, conversational security, conversational help, conversational prioritization, conversational resource management, output formatting and presentation, summarization, conversational delayed actions/agents/memorization, conversational logic, and coordinated interfaces and devices. Such services are provided through API calls via the conversational application API Layer 803. The conversational application and behavior/services manager 815 is responsible for executing all the different functions needed to adapt the UI to the capabilities and constraints of the device, application and/or user preferences. To the extent it is found that Maes does not expressly disclose this limitation, the limitation is inherent. And to the extent it is found that Maes does not anticipate this claim, Maes renders it obvious, standing alone, in combination with the knowledge of a person of ordinary skill in the art, and/or in combination with one or more other prior art references identified in the Cover Pleading or herein, for the reasons identified herein or in the Cover Pleading. A person of ordinary skill in the art would have been motivated to combine Maes with any of the other prior art references identified in the Cover Pleading or herein with a reasonable expectation of success because each of these references contains familiar elements within the similar fields of endeavor to address similar known problems, according to known methods which would yield predictable results.
[g] identifying, based on the one or more rank scores, the one or more context entries	Maes discloses “identifying, based on the one or more rank scores, the one or more context entries from among the plurality of context entries[.]” *E.g.*, Maes at Fig. 2:

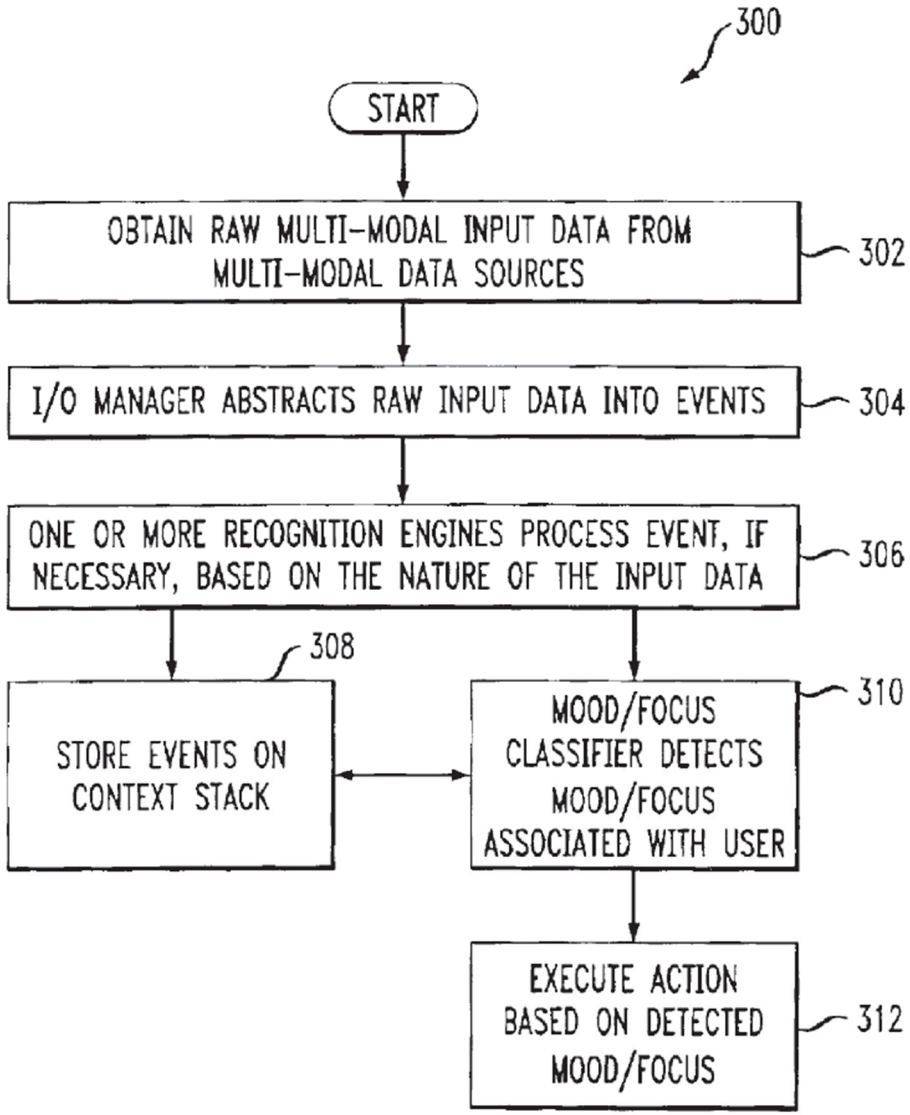
Claim 7 of the ’957 Patent	**Exemplary Citations to Maes**
from among the plurality of context entries; and	

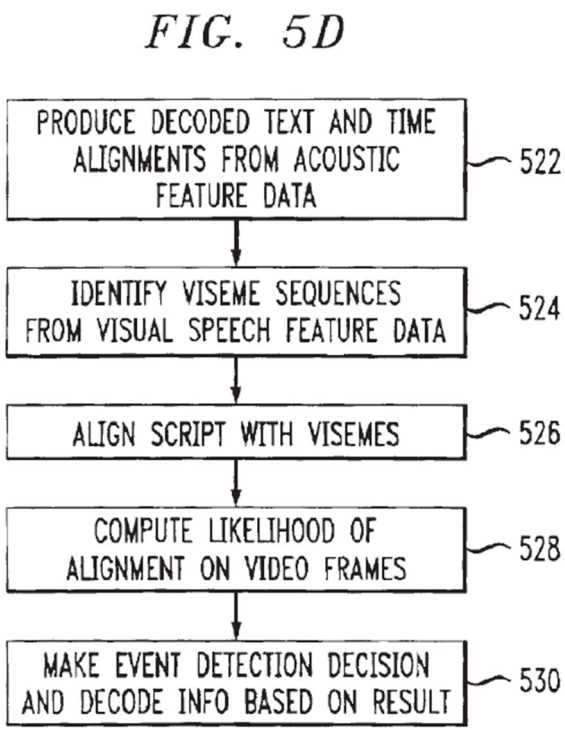
FIG. 2

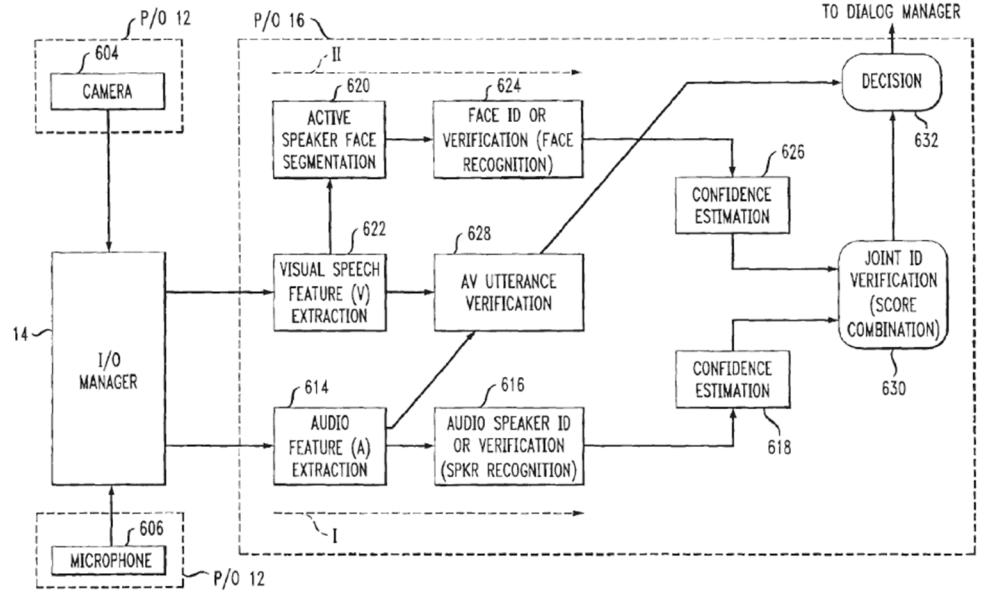

200
START
OBTAIN RAW MULTI-MODAL INPUT DATA FROM MULTI-MODAL DATA SOURCES
202
I/O MANAGER ABSTRACTS RAW INPUT DATA INTO EVENTS
204
ONE OR MORE RECOGNITION ENGINES PROCESS EVENT, IF NECESSARY, BASED ON THE NATURE OF THE INPUT DATA
206
208
STORE EVENTS ON CONTEXT STACK
DIALOG MANAGER ATTEMPTS TO DETERMINE INTENT OF USER BASED ON PROCESSED ABSTRACT EVENT AND CONTEXT STACK HISTORY
210
212
NO
INTENT KNOWN ?
YES
214
DIALOG MANAGER CAUSES GENERATION OF OUTPUT TO USER TO OBTAIN FURTHER INPUT DATA
216
NO
CONFIRM INTENT ?
YES
EXECUTE PROCESSED EVENT OR OTHER ACTION
218

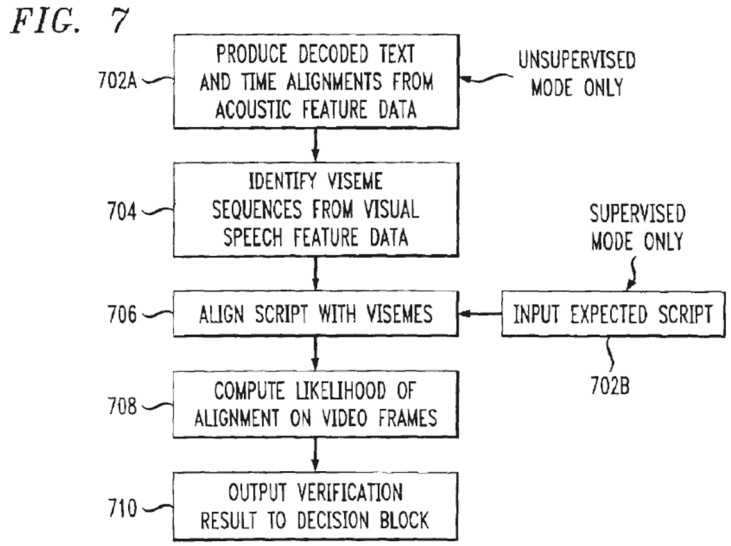
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Maes at Fig. 3:

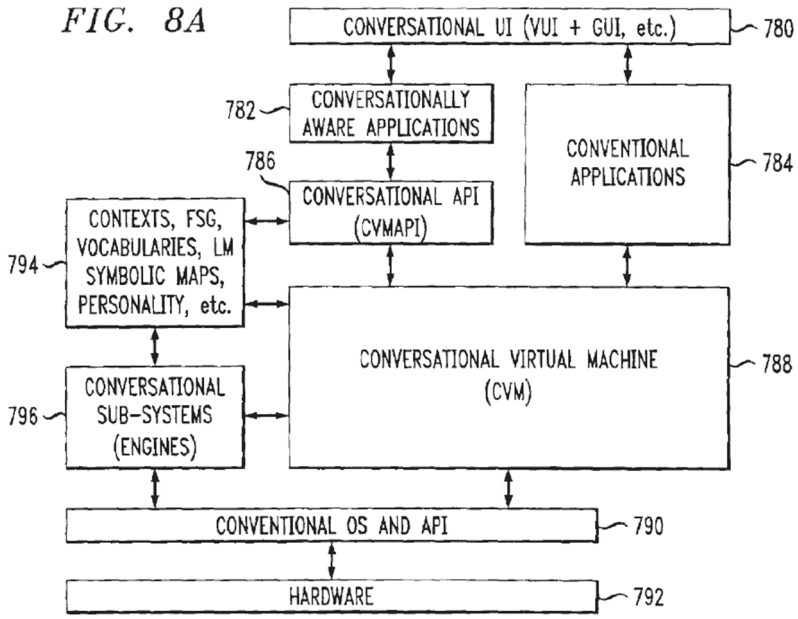
FIG. 3

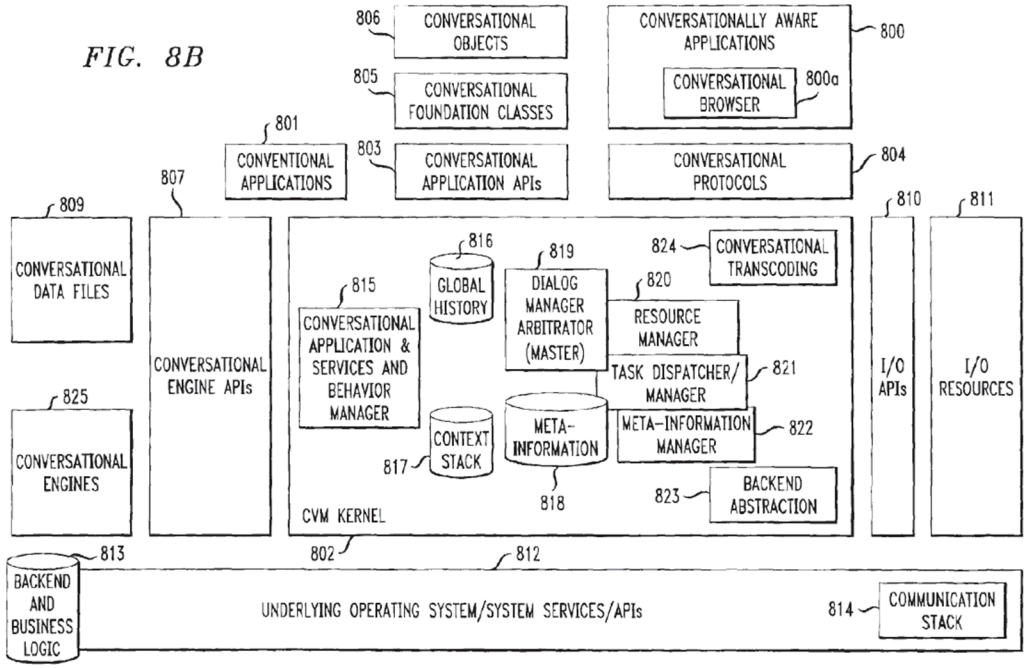

300
START
OBTAIN RAW MULTI-MODAL INPUT DATA FROM MULTI-MODAL DATA SOURCES
302
I/O MANAGER ABSTRACTS RAW INPUT DATA INTO EVENTS
304
ONE OR MORE RECOGNITION ENGINES PROCESS EVENT, IF NECESSARY, BASED ON THE NATURE OF THE INPUT DATA
306
308
STORE EVENTS ON CONTEXT STACK
MOOD/FOCUS CLASSIFIER DETECTS MOOD/FOCUS ASSOCIATED WITH USER
310
EXECUTE ACTION BASED ON DETECTED MOOD/FOCUS
312

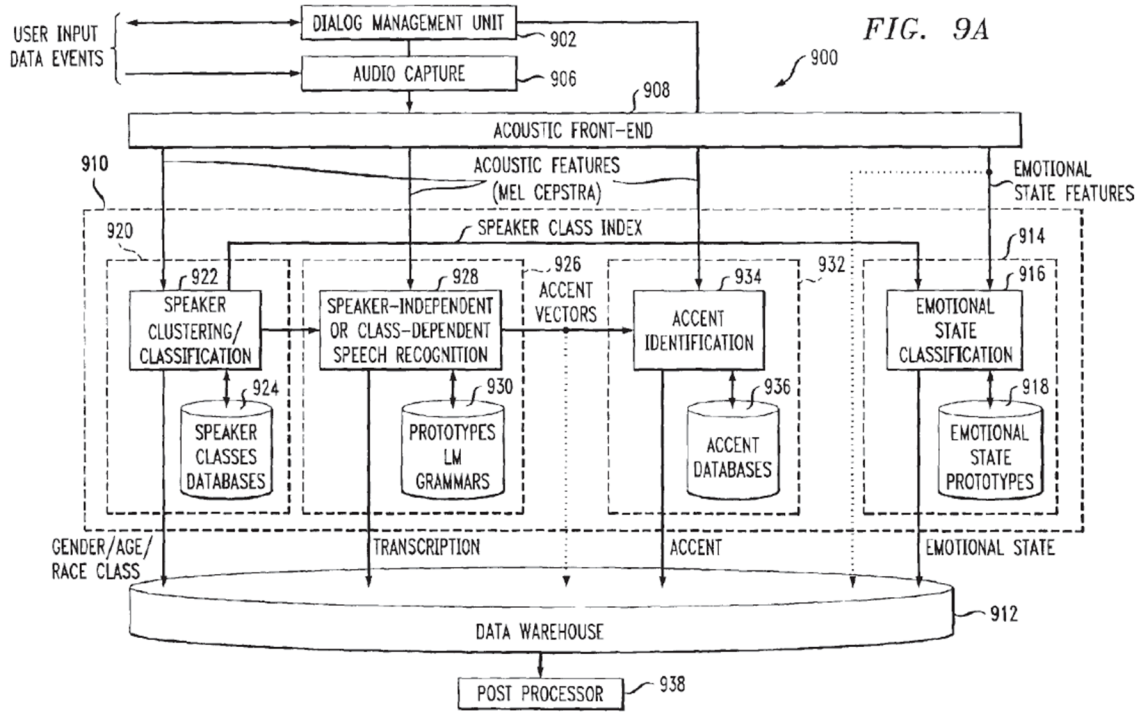
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	Maes at 5D:  Maes at Fig. 6:

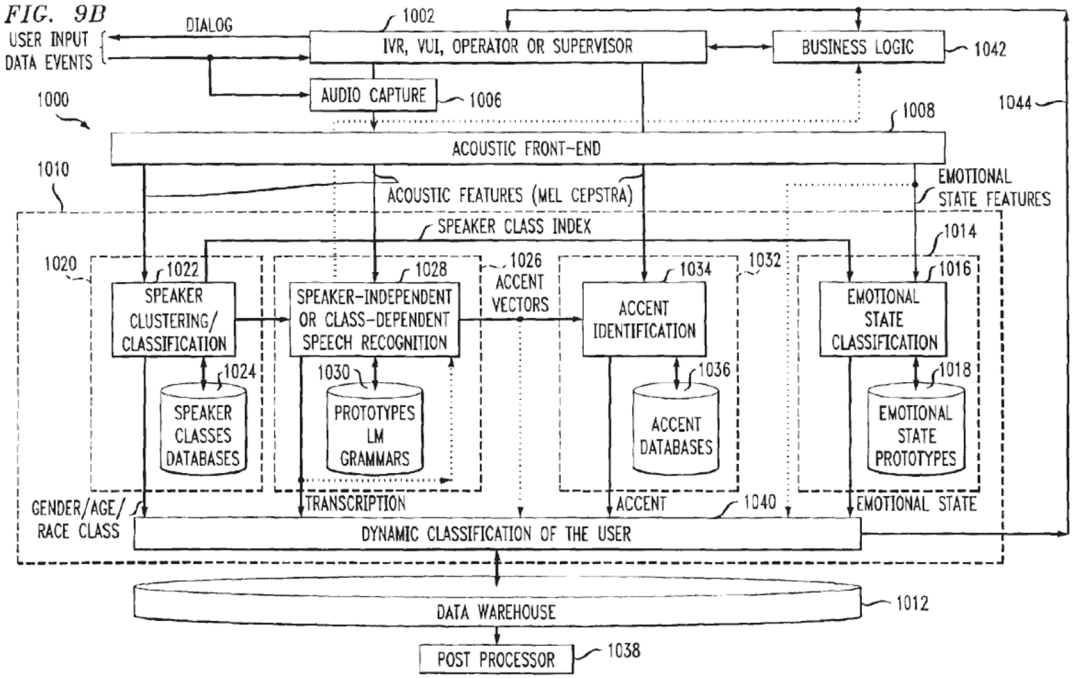
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	FIG. 6  Maes at Fig. 7:

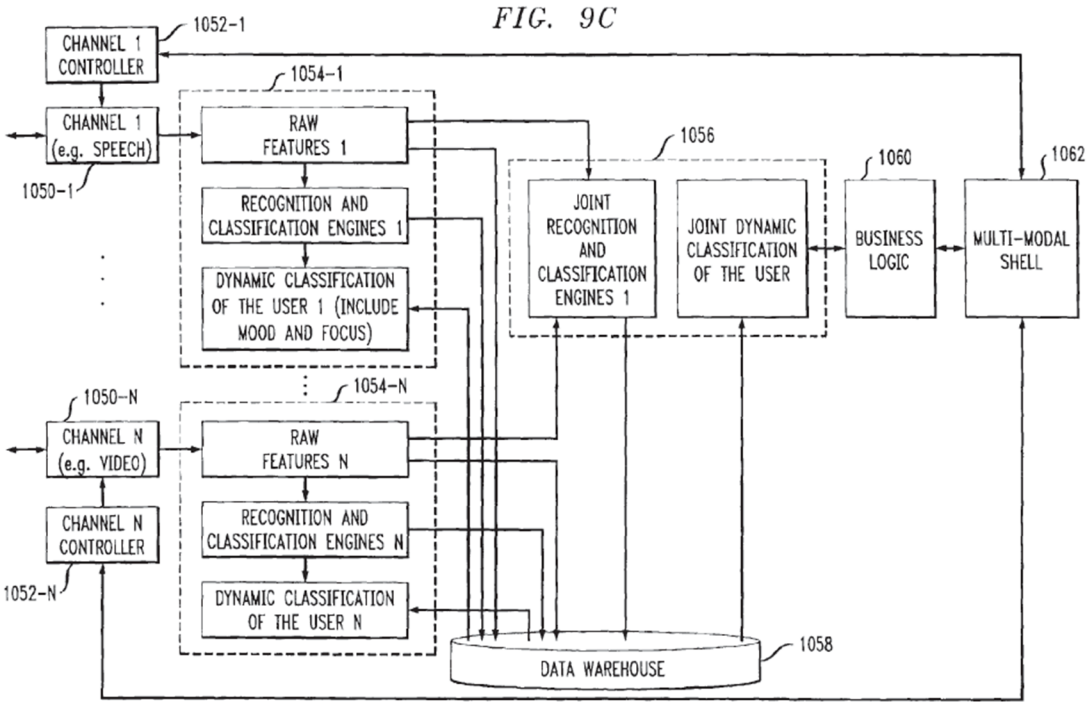
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 8A:

Claim 7 of the '957 Patent	Exemplary Citations to Maes
	 Maes at Fig 8B:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 9A:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 9B:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 9C:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at 7:60-8:65: Thus, the dialog manager must first determine the user's intent based on the current event and, if available, the historical information (e.g., past events) stored in the context stack. For instance, returning to the vehicle example, the user may say “turn it on,” while pointing at the vehicle radio. The dialog manager would therefore receive the results of the recognized events associated with the spoken utterance “turn it on” and the gesture of pointing to the radio. Based on these events, the dialog manager does a search of the existing applications, transactions or “dialogs,” or portions thereof, with which such an utterance and gesture could be associated. Accordingly, as shown in FIG. 1, the dialog manager 18 determines the appropriate CML-authored application 24. The

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	application may be stored on the system 10 or accessed (e.g., downloaded) from some remote location. If the dialog manager determines with some predetermined degree of confidence that the application it selects is the one which will effectuate the users desire, the dialog manager executes the next step of the multi-modal dialog (e.g., prompt or display for missing, ambiguous or confusing information, asks for confirmation or launches the execution of an action associated to a fully understood multi-modal request from the user) of that application based on the multi-modal input. That is, the dialog manager selects the appropriate device (e.g., radio) activation routine and instructs the I/O manager to output a command to activate the radio. The predetermined degree of confidence may be that at least two input parameters or variables of the application are satisfied or provided by the received events. Of course, depending on the application, other levels of confidence and algorithms may be established as, for example, described in K. A. Papineni, S. Roukos, R. T. Ward, “Free-flow dialog management using forms,” Proc. Eurospeech, Budapest, 1999; and K. Davies et al., “The IBM conversational telephony system for financial applications,” Proc. Eurospeech, Budapest, 1999, the disclosures of which are incorporated by reference herein. Consider the case where the user first says “turn it on,” and then a few seconds later points to the radio. The dialog manager would first try to determine user intent based solely on the “turn it on” command. However, since there are likely other devices in the vehicle that could be turned on, the system would likely not be able to determine with a sufficient degree of confidence what the user was referring to. However, this recognized spoken utterance event is stored on the context stack. Then, when the recognized gesture event (e.g., pointing to the radio) is received, the dialog manager takes this event and the previous spoken utterance event stored on the context stack and makes a determination that the user intended to have the radio turned on. Consider the case where the user says “turn it on,” but makes no gesture and provides no other utterance. In this case, assume that the dialog manager does not have enough input to determine the user intent (step 212 in FIG. 2) and thus implement the command. The dialog manager, in step 214, then causes the generation of an output to the user requesting further input data so that the user's intent can be disambiguated. This may be accomplished by the dialog manager instructing the I/O manager to have the I/O subsystem output a request for clarification. In one embodiment, the I/O subsystem 12 may comprise a text-to-speech (TTS) engine and one or more output

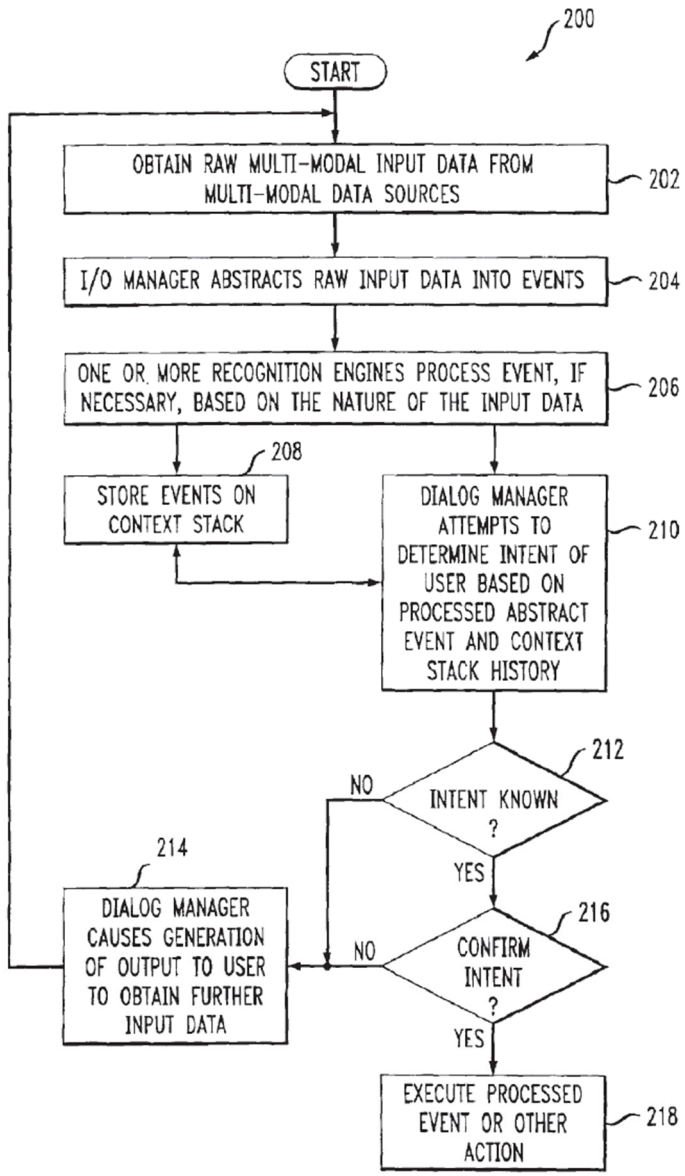
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	speakers. The dialog manager then generates a predetermined question such as “what device do you want to have turned on?” which the TTS engine converts to a synthesized utterance that is audibly output by the speaker to the user. The user, hearing the query, could then point to the radio or say “the radio” thereby providing the dialog manager with the additional input data to disambiguate his request. That is, with reference to FIG. 2, the system 10 obtains the raw input data, again in step 202, and the process 200 iterates based on the new data. Such iteration can continue as long as necessary for the dialog manager to determine the user's intent. Maes at 20:11-20: Then, a search is performed in search module 434 with language models (LM) based on the weighted probabilities received from module 430. That is, the acoustic units identified as having the highest probabilities of representing what was uttered in the arbitrary content video are put together to form words. The words are output by the search engine 434 as the decoded system output. A conventional search engine may be employed. This output is provided to the dialog manager 18 of FIG. 1 for use in disambiguating the user's intent, as described above. Maes at 33:26-34:11: Referring now to FIG. 8B, a diagram illustrates abstract programming layers of a CVM according to a preferred embodiment. The abstract layers of the CVM comprise conversationally aware applications 800 and conventional applications 801 that can run on top of the CVM. An application that relies on multi-modal disambiguation is an example of such a conversational application that executes on top of the CVM. Similarly, an application that exploits focus information or mood can be considered as a conversational application on top of the CVM. These applications are the programs that are executed by the system to provide the user with the interaction he desires within the environment in which the system is deployed. As discussed above, the conversationally aware applications 800 interact with a CVM kernel layer 802 via a conversational application API layer 803. The conversational application API layer 803 encompasses conversational programming languages/scripts and libraries (conversational foundation classes) to provide the various features offered by the CVM kernel 802. For example,

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	the conversational programming languages/scripts provide the conversational APIs that allow an application developer to hook (or develop) conversationally aware applications 800. They also provide the conversational API layer 803, conversational protocols 804 and system calls that allow a developer to build the conversational features into an application to make it “conversationally aware.” The code implementing the applications, API calls and protocol calls includes interpreted and compiled scripts and programs, with library links, conversational logic engine call and conversational foundation classes. More specifically, the conversational application API layer 803 comprises a plurality of conversational foundation classes 805 (or fundamental dialog components) which are provided to the application developer through library functions that may be used to build a CUI or conversationally aware applications 800. The conversational foundation classes 805 are the elementary components or conversational gestures (as described by T. V. Raman, in “Auditory User Interfaces, Toward The Speaking Computer,” Kluwer Academic Publishers, Boston 1997) that characterize any dialog, independently of the modality or combination of modalities (which can be implemented procedurally or declaratively). The conversational foundation classes 805 comprise CUI building blocks and conversational platform libraries, dialog modules and components, and dialog scripts and beans. The conversational foundation classes 805 may be compiled locally into conversational objects 806. More specifically, the conversational objects 805 (or dialog components) are compiled from the conversational foundation classes 805 (fundamental dialog components) by combining the different individual classes in a code calling these libraries through a programming language such as Java or C++. Maes at 36:8-48: The core CVM kernel layer 802 comprises programming layers such as a conversational application and behavior/service manager layer 815, a conversational dialog manager (arbitrator) layer 819, a conversational resource manager layer 820, a task/dispatcher manager 821 and a meta-information manager 822, which provide the core functions of the CVM layer 802. It is to be understood that these components may be implemented as part of the dialog manager 18 of FIG. 1. The conversational application and behavior/service manager layer 815 comprises functions for

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	managing the conventional and conversationally aware applications 800 and 801. Such management functions include, for example, keeping track of which applications are registered (both local and network-distributed), what are the dialog interfaces (if any) of the applications, and what is the state of each application. In addition, the conversational application and services/behavior manager 815 initiates all the tasks associated with any specific service or behavior provided by the CVM system. The conversational services and behaviors are all the behaviors and features of a conversational UI that the user may expect to find in the applications and interactions, as well as the features that an application developer may expect to be able to access via APIs (without having to implement with the development of the application). Examples of the conversational services and behavior provided by the CVM kernel 802 include, but are not limited to, conversational categorization and meta-information, conversational object, resource and file management, conversational search, conversational selection, conversational customization, conversational security, conversational help, conversational prioritization, conversational resource management, output formatting and presentation, summarization, conversational delayed actions/agents/memorization, conversational logic, and coordinated interfaces and devices. Such services are provided through API calls via the conversational application API Layer 803. The conversational application and behavior/services manager 815 is responsible for executing all the different functions needed to adapt the UI to the capabilities and constraints of the device, application and/or user preferences. Maes at 38:3-17: The CVM kernel layer 802 further comprises a backend abstraction layer 823 which allows access to backend business logic 813 via the dialog manager 819 (rather than bypassing the dialog manager 819). This allows such accesses to be added to the context stack 817 and global history 816. For instance, the backend abstraction layer 823 can translate input and output to and from the dialog manager 819 to database queries. This layer 823 will convert standardized attribute value n-tuples into database queries and translate the result of such queries into tables or sets of attribute value n-tuples back to the dialog manager 819. In addition, a conversational transcoding layer 824 is provided to adapt the behavior, UI and dialog presented to the user based on the I/O and engine capabilities of the device which executes the CVM system.

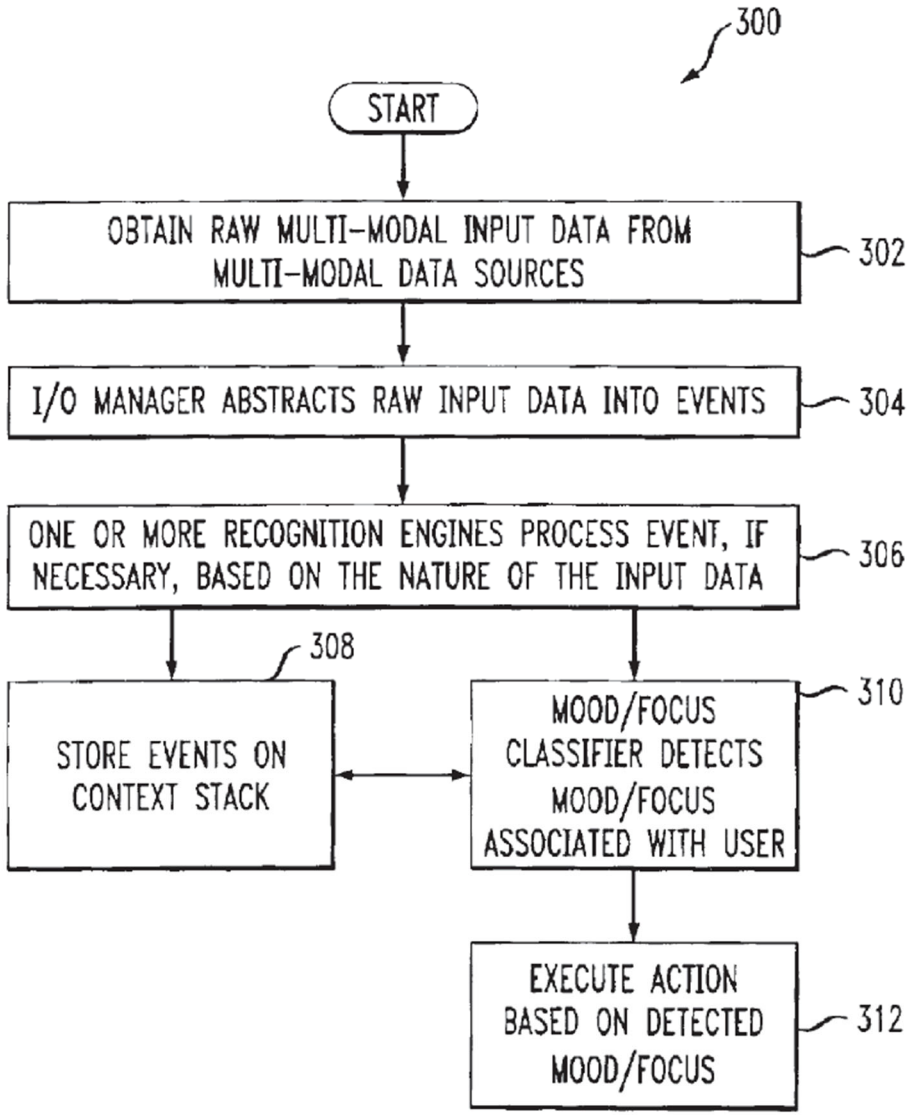
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	To the extent it is found that Maes does not expressly disclose this limitation, the limitation is inherent. And to the extent it is found that Maes does not anticipate this claim, Maes renders it obvious, standing alone, in combination with the knowledge of a person of ordinary skill in the art, and/or in combination with one or more other prior art references identified in the Cover Pleading or herein, for the reasons identified herein or in the Cover Pleading. A person of ordinary skill in the art would have been motivated to combine Maes with any of the other prior art references identified in the Cover Pleading or herein with a reasonable expectation of success because each of these references contains familiar elements within the similar fields of endeavor to address similar known problems, according to known methods which would yield predictable results.
[h] determining based on the determined one or more words and the context information, the command or the request associated with the natural language utterance	Maes discloses “determining based on the determined one or more words and the context information, the command or the request associated with the natural language utterance[.]” *E.g.*, Maes at Fig. 2:

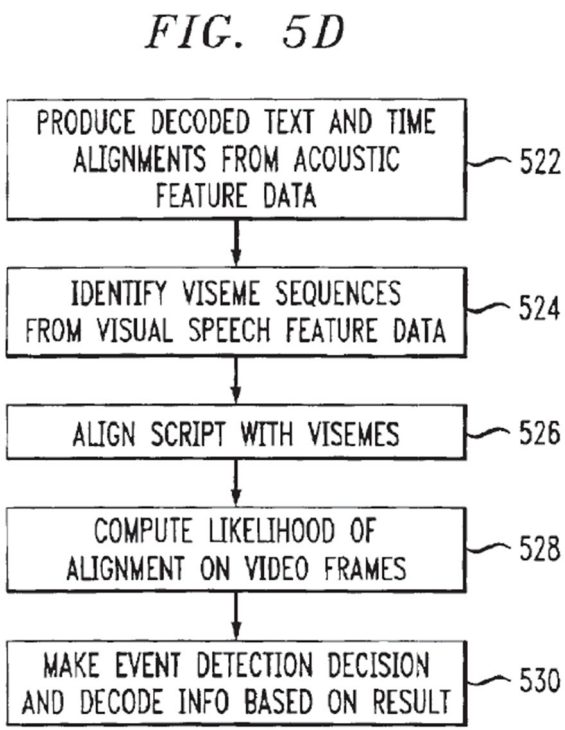
FIG. 2

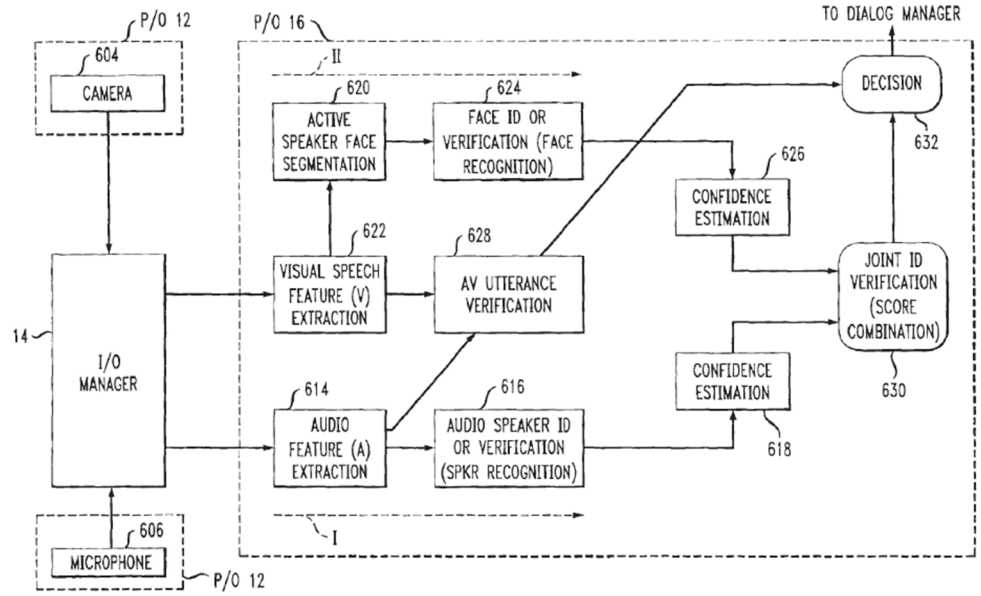

200
START
OBTAIN RAW MULTI-MODAL INPUT DATA FROM MULTI-MODAL DATA SOURCES
202
I/O MANAGER ABSTRACTS RAW INPUT DATA INTO EVENTS
204
ONE OR MORE RECOGNITION ENGINES PROCESS EVENT, IF NECESSARY, BASED ON THE NATURE OF THE INPUT DATA
206
208
STORE EVENTS ON CONTEXT STACK
DIALOG MANAGER ATTEMPTS TO DETERMINE INTENT OF USER BASED ON PROCESSED ABSTRACT EVENT AND CONTEXT STACK HISTORY
210
212
NO
INTENT KNOWN ?
YES
214
DIALOG MANAGER CAUSES GENERATION OF OUTPUT TO USER TO OBTAIN FURTHER INPUT DATA
216
NO
CONFIRM INTENT ?
YES
EXECUTE PROCESSED EVENT OR OTHER ACTION
218

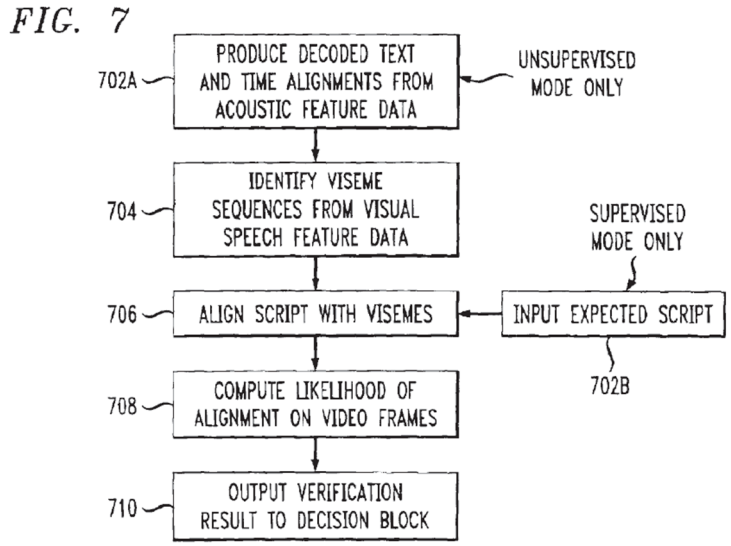
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Maes at Fig. 3:

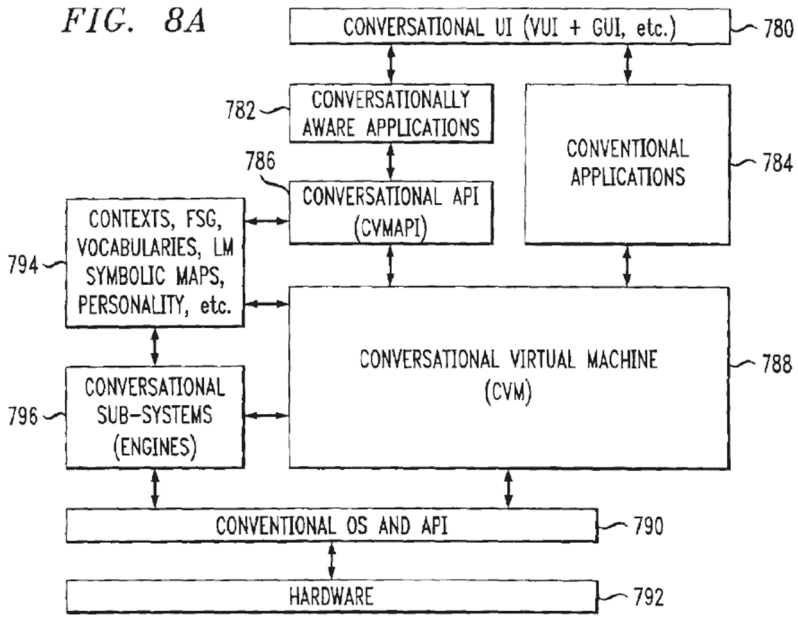
FIG. 3

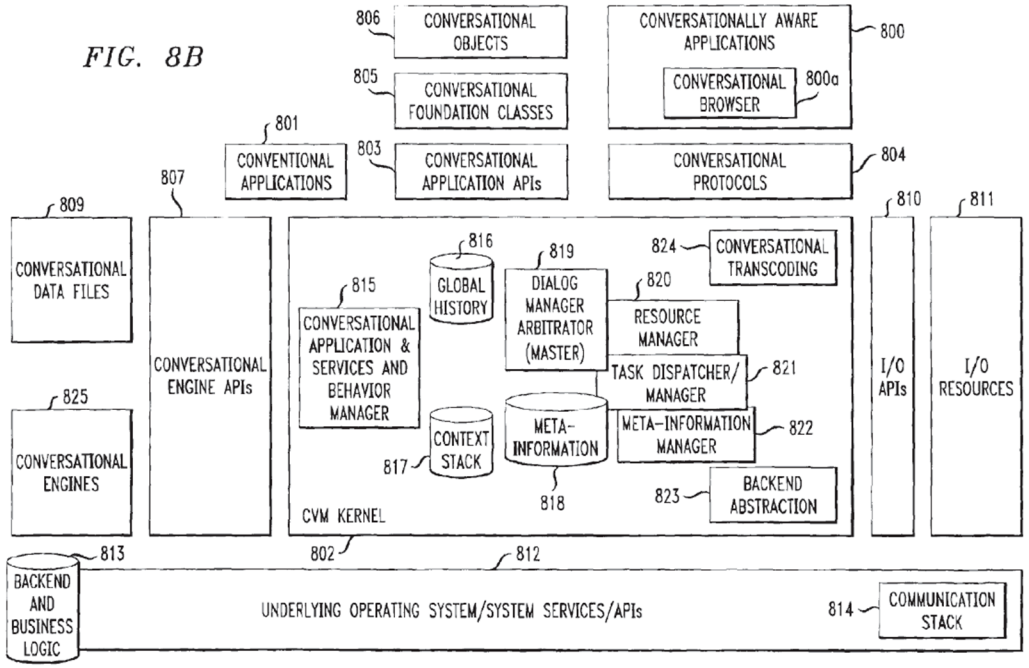

300
START
OBTAIN RAW MULTI-MODAL INPUT DATA FROM MULTI-MODAL DATA SOURCES
302
I/O MANAGER ABSTRACTS RAW INPUT DATA INTO EVENTS
304
ONE OR MORE RECOGNITION ENGINES PROCESS EVENT, IF NECESSARY, BASED ON THE NATURE OF THE INPUT DATA
306
308
STORE EVENTS ON CONTEXT STACK
MOOD/FOCUS CLASSIFIER DETECTS MOOD/FOCUS ASSOCIATED WITH USER
310
EXECUTE ACTION BASED ON DETECTED MOOD/FOCUS
312

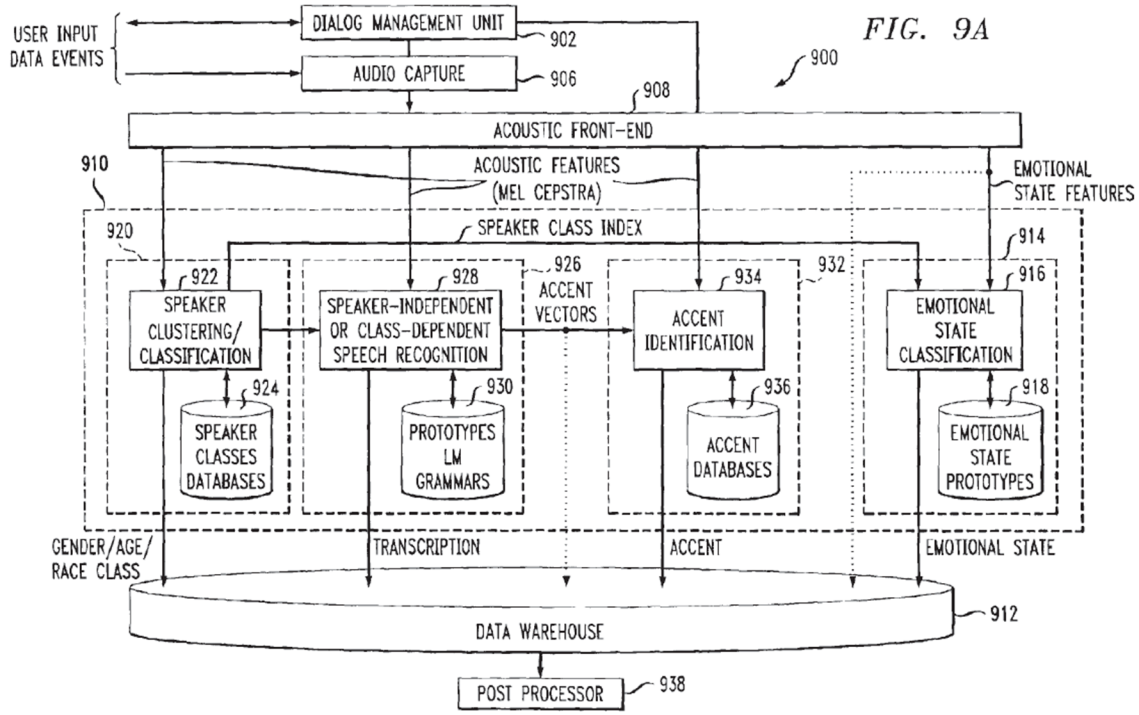
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	Maes at 5D:  Maes at Fig. 6:

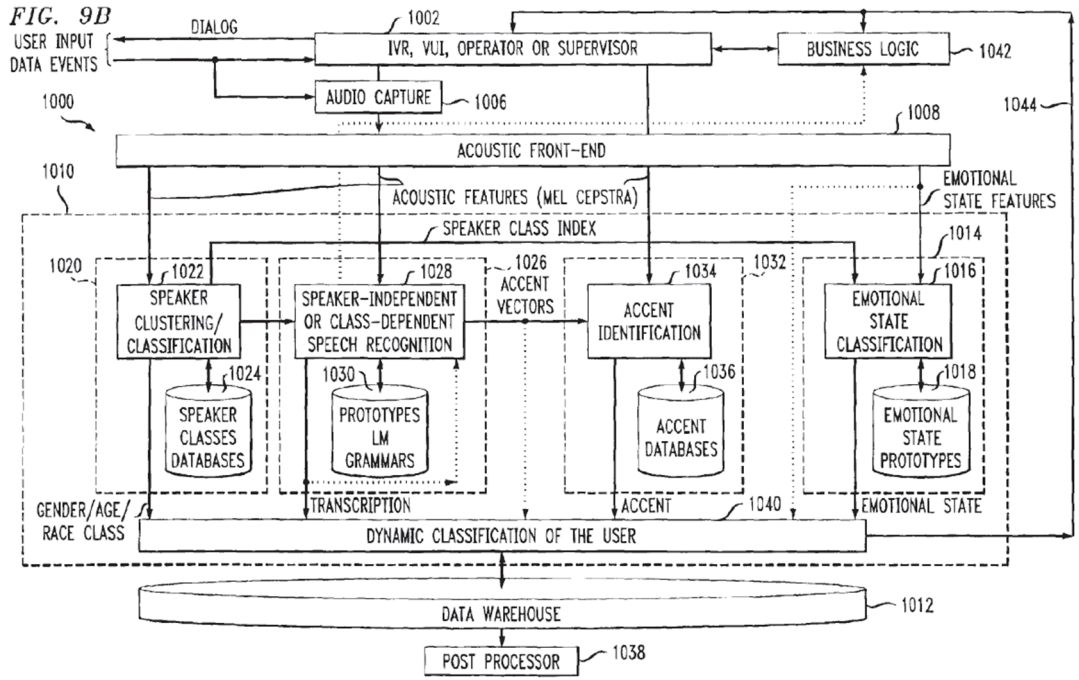
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	FIG. 6  Maes at Fig. 7:

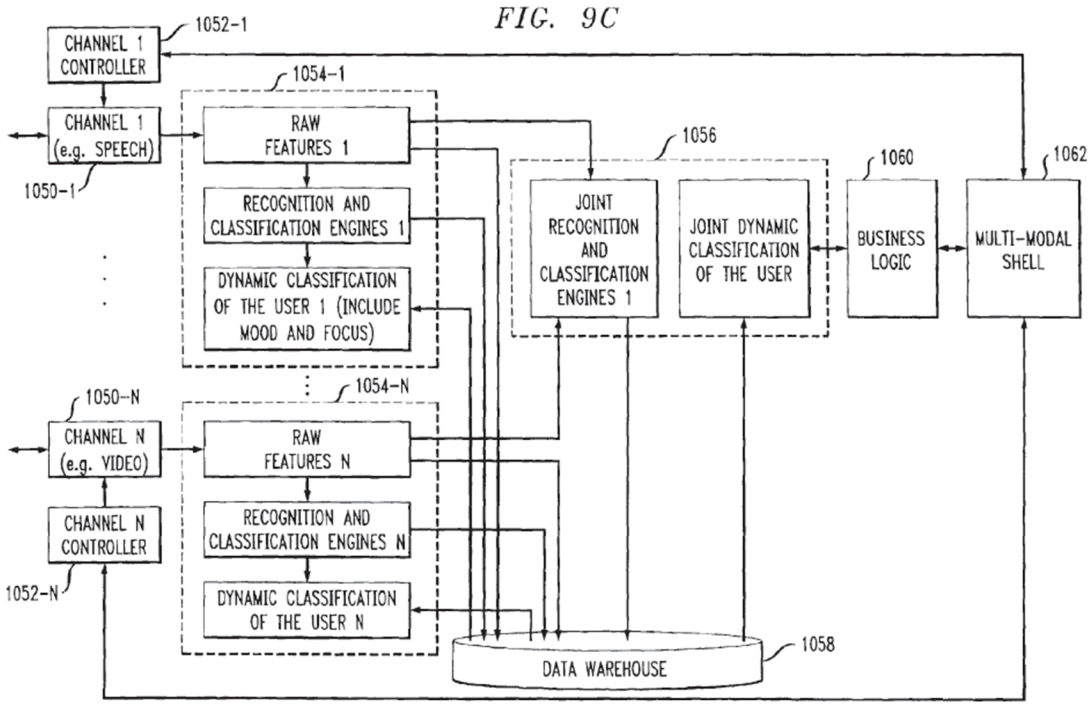
Claim 7 of the '957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 8A:

Claim 7 of the '957 Patent	Exemplary Citations to Maes
	 Maes at Fig 8B:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 9A:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 9B:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 9C:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at 7:33-45: In step 208, the recognized events, as well as the events that do not need to be recognized, are stored in a storage unit referred to as the context stack 20. The context stack is used to create a history of interaction between the user and the system so as to assist the dialog manager 18 in making referential ambiguity resolution determinations when determining the user's intent. Next, in step 210, the system 10 attempts to determine the user intent based on the current event and the historical interaction information stored in the context stack and then determine and execute one or more application programs that effectuate the user's intention and/or react to the user activity.

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Maes at 7:60-8:65: Thus, the dialog manager must first determine the user's intent based on the current event and, if available, the historical information (e.g., past events) stored in the context stack. For instance, returning to the vehicle example, the user may say “turn it on,” while pointing at the vehicle radio. The dialog manager would therefore receive the results of the recognized events associated with the spoken utterance “turn it on” and the gesture of pointing to the radio. Based on these events, the dialog manager does a search of the existing applications, transactions or “dialogs,” or portions thereof, with which such an utterance and gesture could be associated. Accordingly, as shown in FIG. 1, the dialog manager 18 determines the appropriate CML-authored application 24. The application may be stored on the system 10 or accessed (e.g., downloaded) from some remote location. If the dialog manager determines with some predetermined degree of confidence that the application it selects is the one which will effectuate the users desire, the dialog manager executes the next step of the multi-modal dialog (e.g., prompt or display for missing, ambiguous or confusing information, asks for confirmation or launches the execution of an action associated to a fully understood multi-modal request from the user) of that application based on the multi-modal input. That is, the dialog manager selects the appropriate device (e.g., radio) activation routine and instructs the I/O manager to output a command to activate the radio. The predetermined degree of confidence may be that at least two input parameters or variables of the application are satisfied or provided by the received events. Of course, depending on the application, other levels of confidence and algorithms may be established as, for example, described in K. A. Papineni, S. Roukos, R. T. Ward, “Free-flow dialog management using forms,” Proc. Eurospeech, Budapest, 1999; and K. Davies et al., “The IBM conversational telephony system for financial applications,” Proc. Eurospeech, Budapest, 1999, the disclosures of which are incorporated by reference herein. Consider the case where the user first says “turn it on,” and then a few seconds later points to the radio. The dialog manager would first try to determine user intent based solely on the “turn it on” command. However, since there are likely other devices in the vehicle that could be turned on, the system would likely not be able to determine with a sufficient degree of confidence what the user was referring to. However, this recognized spoken utterance event is stored on the context stack.

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Then, when the recognized gesture event (e.g., pointing to the radio) is received, the dialog manager takes this event and the previous spoken utterance event stored on the context stack and makes a determination that the user intended to have the radio turned on. Consider the case where the user says “turn it on,” but makes no gesture and provides no other utterance. In this case, assume that the dialog manager does not have enough input to determine the user intent (step 212 in FIG. 2) and thus implement the command. The dialog manager, in step 214, then causes the generation of an output to the user requesting further input data so that the user's intent can be disambiguated. This may be accomplished by the dialog manager instructing the I/O manager to have the I/O subsystem output a request for clarification. In one embodiment, the I/O subsystem 12 may comprise a text-to-speech (TTS) engine and one or more output speakers. The dialog manager then generates a predetermined question such as “what device do you want to have turned on?” which the TTS engine converts to a synthesized utterance that is audibly output by the speaker to the user. The user, hearing the query, could then point to the radio or say “the radio” thereby providing the dialog manager with the additional input data to disambiguate his request. That is, with reference to FIG. 2, the system 10 obtains the raw input data, again in step 202, and the process 200 iterates based on the new data. Such iteration can continue as long as necessary for the dialog manager to determine the user's intent. Maes at 20:11-20: Then, a search is performed in search module 434 with language models (LM) based on the weighted probabilities received from module 430. That is, the acoustic units identified as having the highest probabilities of representing what was uttered in the arbitrary content video are put together to form words. The words are output by the search engine 434 as the decoded system output. A conventional search engine may be employed. This output is provided to the dialog manager 18 of FIG. 1 for use in disambiguating the user's intent, as described above. Maes at 33:26-34:11:

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Referring now to FIG. 8B, a diagram illustrates abstract programming layers of a CVM according to a preferred embodiment. The abstract layers of the CVM comprise conversationally aware applications 800 and conventional applications 801 that can run on top of the CVM. An application that relies on multi-modal disambiguation is an example of such a conversational application that executes on top of the CVM. Similarly, an application that exploits focus information or mood can be considered as a conversational application on top of the CVM. These applications are the programs that are executed by the system to provide the user with the interaction he desires within the environment in which the system is deployed. As discussed above, the conversationally aware applications 800 interact with a CVM kernel layer 802 via a conversational application API layer 803. The conversational application API layer 803 encompasses conversational programming languages/scripts and libraries (conversational foundation classes) to provide the various features offered by the CVM kernel 802. For example, the conversational programming languages/scripts provide the conversational APIs that allow an application developer to hook (or develop) conversationally aware applications 800. They also provide the conversational API layer 803, conversational protocols 804 and system calls that allow a developer to build the conversational features into an application to make it “conversationally aware.” The code implementing the applications, API calls and protocol calls includes interpreted and compiled scripts and programs, with library links, conversational logic engine call and conversational foundation classes. More specifically, the conversational application API layer 803 comprises a plurality of conversational foundation classes 805 (or fundamental dialog components) which are provided to the application developer through library functions that may be used to build a CUI or conversationally aware applications 800. The conversational foundation classes 805 are the elementary components or conversational gestures (as described by T. V. Raman, in “Auditory User Interfaces, Toward The Speaking Computer,” Kluwer Academic Publishers, Boston 1997) that characterize any dialog, independently of the modality or combination of modalities (which can be implemented procedurally or declaratively). The conversational foundation classes 805 comprise CUI building blocks and conversational platform libraries, dialog modules and components, and dialog scripts and beans. The conversational foundation classes 805 may be compiled locally into conversational objects 806. More specifically, the conversational objects 805

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	(or dialog components) are compiled from the conversational foundation classes 805 (fundamental dialog components) by combining the different individual classes in a code calling these libraries through a programming language such as Java or C++. Maes at 36:8-48: The core CVM kernel layer 802 comprises programming layers such as a conversational application and behavior/service manager layer 815, a conversational dialog manager (arbitrator) layer 819, a conversational resource manager layer 820, a task/dispatcher manager 821 and a meta-information manager 822, which provide the core functions of the CVM layer 802. It is to be understood that these components may be implemented as part of the dialog manager 18 of FIG. 1. The conversational application and behavior/service manager layer 815 comprises functions for managing the conventional and conversationally aware applications 800 and 801. Such management functions include, for example, keeping track of which applications are registered (both local and network-distributed), what are the dialog interfaces (if any) of the applications, and what is the state of each application. In addition, the conversational application and services/behavior manager 815 initiates all the tasks associated with any specific service or behavior provided by the CVM system. The conversational services and behaviors are all the behaviors and features of a conversational UI that the user may expect to find in the applications and interactions, as well as the features that an application developer may expect to be able to access via APIs (without having to implement with the development of the application). Examples of the conversational services and behavior provided by the CVM kernel 802 include, but are not limited to, conversational categorization and meta-information, conversational object, resource and file management, conversational search, conversational selection, conversational customization, conversational security, conversational help, conversational prioritization, conversational resource management, output formatting and presentation, summarization, conversational delayed actions/agents/memorization, conversational logic, and coordinated interfaces and devices. Such services are provided through API calls via the conversational application API Layer 803. The conversational application and behavior/services manager 815 is responsible for executing all the different functions needed to adapt the UI to the capabilities and constraints of the device, application and/or user preferences.

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	Maes at 36:49-37:3: The conversational dialog manager 819 comprises functions for managing the dialog (conversational dialog comprising speech and other multi-modal I/O such as GUI keyboard, pointer, mouse, as well as video input, etc.) and arbitration (dialog manager arbitrator or DMA) across all registered applications. In particular, the conversational dialog manager 819 determines what information the user has, which inputs the user presents, and which application(s) should handle the user inputs. The DMA processes abstracted I/O events (abstracted by the I/O manager) using the context/history to understand the user intent. When an abstract event occurs, the DMA determines the target of the event and, if needed, seeks confirmation, disambiguation, correction, more details, etc., until the intent is unambiguous and fully determined. The DMA then launches the action associated to the user's query. The DMA function handles multi-modal I/O events to: (1) determine the target application or dialog (or portion of it); and (2) use past history and context to: (a) understand the intent of the user; (b) follow up with a dialog to disambiguate, complete, correct or confirm the understanding; (c) or, dispatch a task resulting from fall understanding of the intent of the user. Maes at 37:17-50: The task dispatcher/manager 821 dispatches and coordinates different tasks and processes that are spawned (by the user and machine) on local and networked conventional and conversational resources. The meta-information manager 822 manages the meta-information associated with the system via a meta-information repository 818. The meta-information manager 822 and repository 818 collect all the information typically assumed known in a conversational interaction but not available at the level of the current conversation. Examples are a-priori knowledge, cultural, educational assumptions and persistent information, past request, references, information about the user, the application, news, etc. It is typically the information that needs to be preserved and persist beyond the length/life of the conversational history/context and the information that is expected to be common knowledge for the conversation and, therefore, has never been defined during the current and possible past conversational interactions. Also, shortcuts to commands,

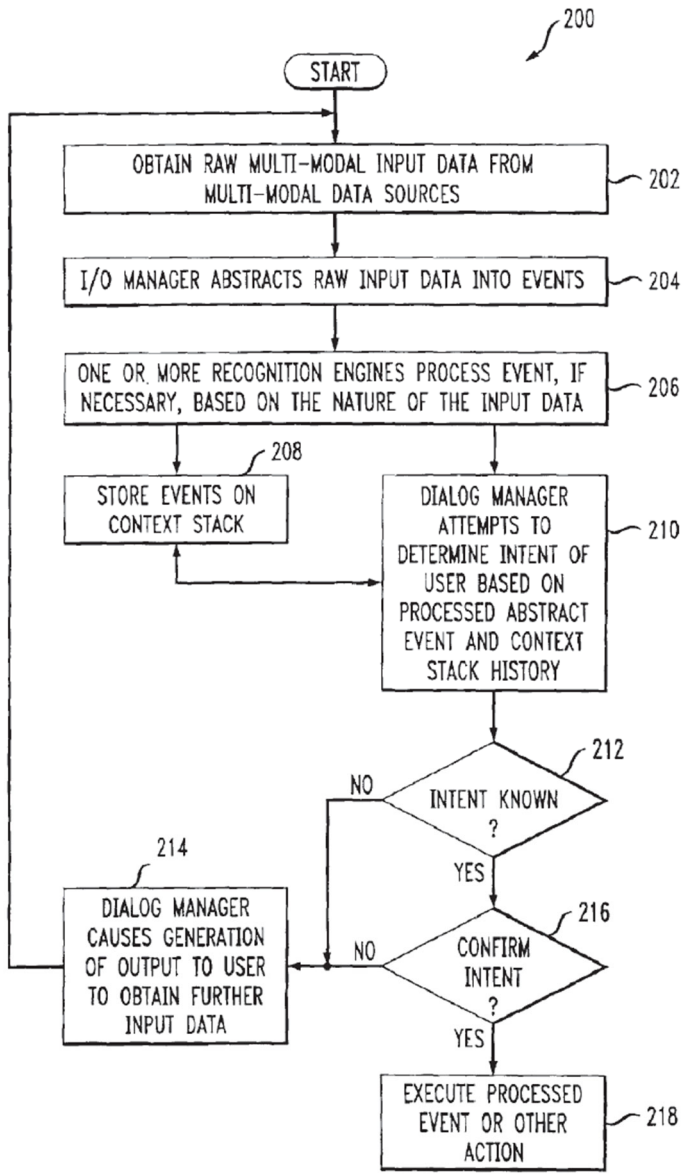
Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	resources and macros, etc., are managed by the meta-information manager 822 and stored in the meta-information repository 818. In addition, the meta-information repository 818 includes a user-usage log based on user identity. It is to be appreciated that services such as conversational help and assistance, as well as some dialog prompts (introduction, questions, feedback, etc.) provided by the CVM system can be tailored based on the usage history of the user as stored in the meta-information repository 818 and associated with the application. If a user has been previously interacting with a given application, an explanation can be reduced assuming that it is familiar to the user. Similarly, if a user commits many errors, the explanations can be more complex, as multiple errors are interpreted as user uncertainty, unfamiliarity, or incomprehension/misunderstanding of the application or function. Maes at 38:3-17: The CVM kernel layer 802 further comprises a backend abstraction layer 823 which allows access to backend business logic 813 via the dialog manager 819 (rather than bypassing the dialog manager 819). This allows such accesses to be added to the context stack 817 and global history 816. For instance, the backend abstraction layer 823 can translate input and output to and from the dialog manager 819 to database queries. This layer 823 will convert standardized attribute value n-tuples into database queries and translate the result of such queries into tables or sets of attribute value n-tuples back to the dialog manager 819. In addition, a conversational transcoding layer 824 is provided to adapt the behavior, UI and dialog presented to the user based on the I/O and engine capabilities of the device which executes the CVM system. To the extent it is found that Maes does not expressly disclose this limitation, the limitation is inherent. And to the extent it is found that Maes does not anticipate this claim, Maes renders it obvious, standing alone, in combination with the knowledge of a person of ordinary skill in the art, and/or in combination with one or more other prior art references identified in the Cover Pleading or herein, for the reasons identified herein or in the Cover Pleading. A person of ordinary skill in the art would have been motivated to combine Maes with any of the other prior art references identified in the Cover Pleading or herein with a reasonable expectation of success because each

Claim 7 of the ’957 Patent	Exemplary Citations to Maes
	of these references contains familiar elements within the similar fields of endeavor to address similar known problems, according to known methods which would yield predictable results.

2. Claim 8

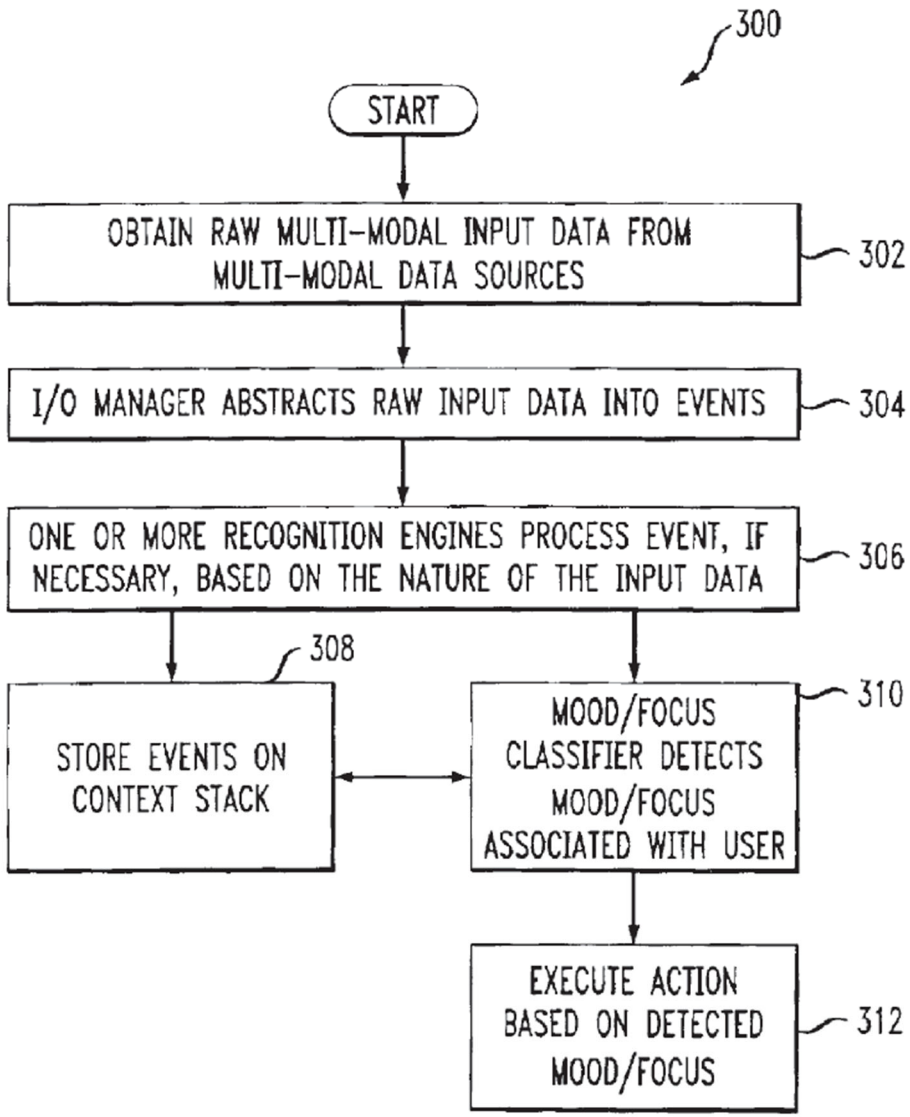
Claim 8 of the ’957 Patent	Exemplary Citations to Maes
[a] The method of claim 7, further comprising: providing a prompt regarding the command or request, wherein the prompt is a prompt for additional information or a prompt for confirmation; and	Maes discloses “providing a prompt regarding the command or request, wherein the prompt is a prompt for additional information or a prompt for confirmation[.]” *E.g.*, Maes at Fig. 2:

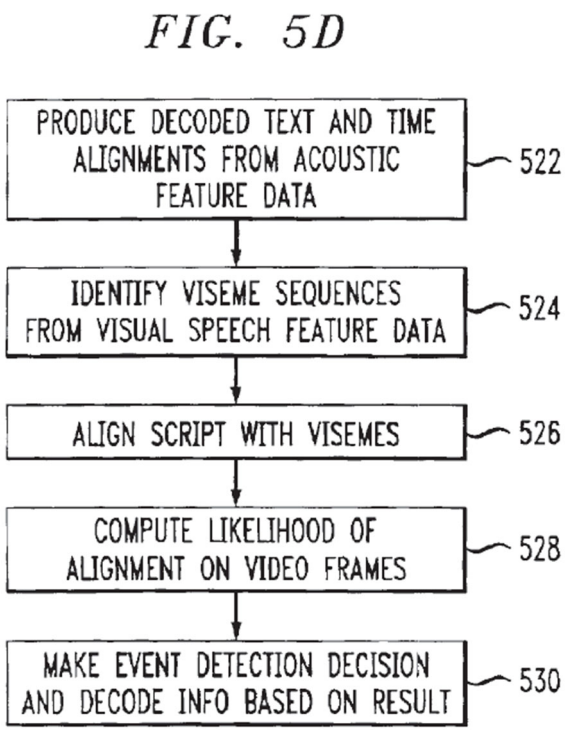
FIG. 2

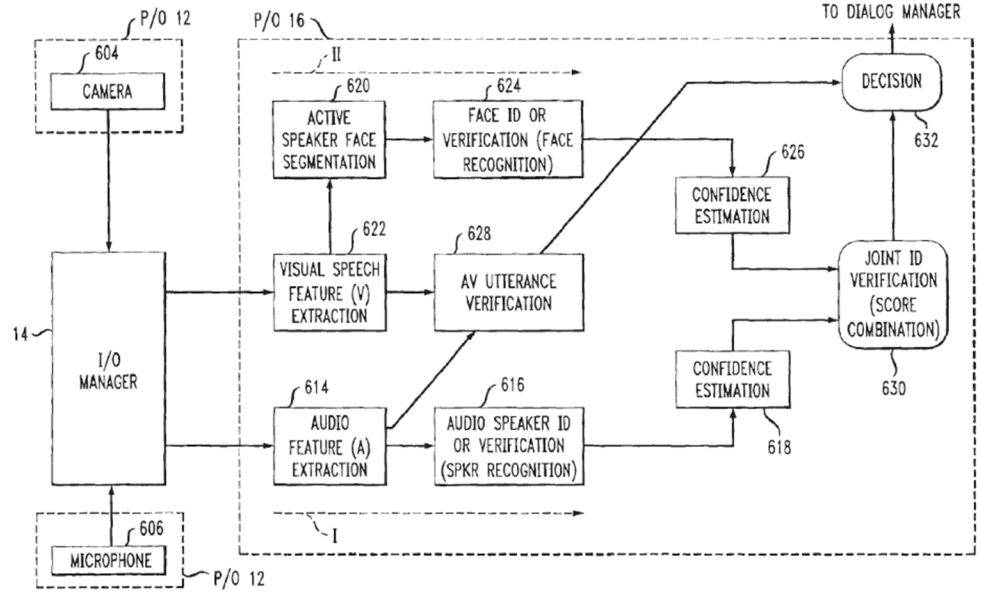

200
START
OBTAIN RAW MULTI-MODAL INPUT DATA FROM MULTI-MODAL DATA SOURCES
202
I/O MANAGER ABSTRACTS RAW INPUT DATA INTO EVENTS
204
ONE OR MORE RECOGNITION ENGINES PROCESS EVENT, IF NECESSARY, BASED ON THE NATURE OF THE INPUT DATA
206
208
STORE EVENTS ON CONTEXT STACK
DIALOG MANAGER ATTEMPTS TO DETERMINE INTENT OF USER BASED ON PROCESSED ABSTRACT EVENT AND CONTEXT STACK HISTORY
210
212
NO
INTENT KNOWN ?
YES
214
DIALOG MANAGER CAUSES GENERATION OF OUTPUT TO USER TO OBTAIN FURTHER INPUT DATA
216
NO
CONFIRM INTENT ?
YES
EXECUTE PROCESSED EVENT OR OTHER ACTION
218

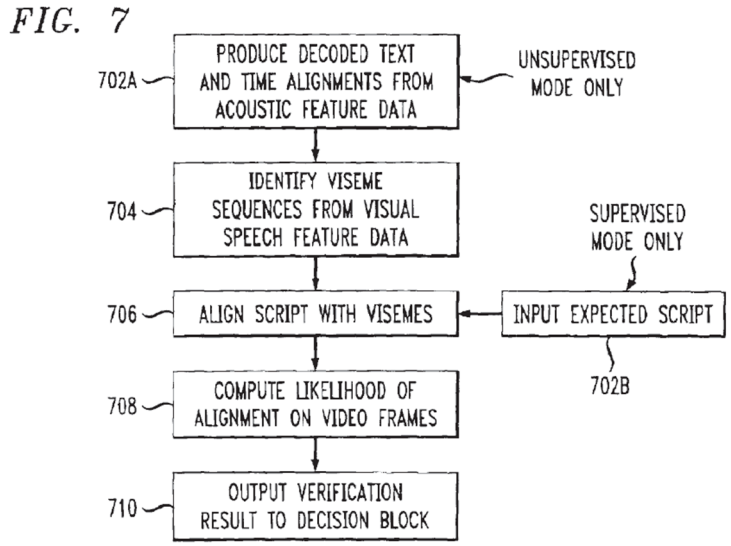
Claim 8 of the ’957 Patent	Exemplary Citations to Maes
	Maes at Fig. 3:

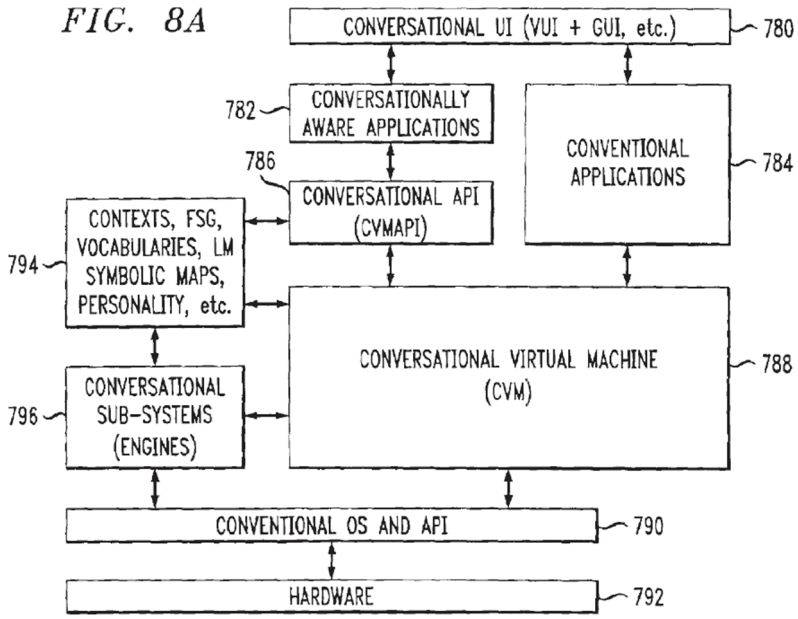
FIG. 3

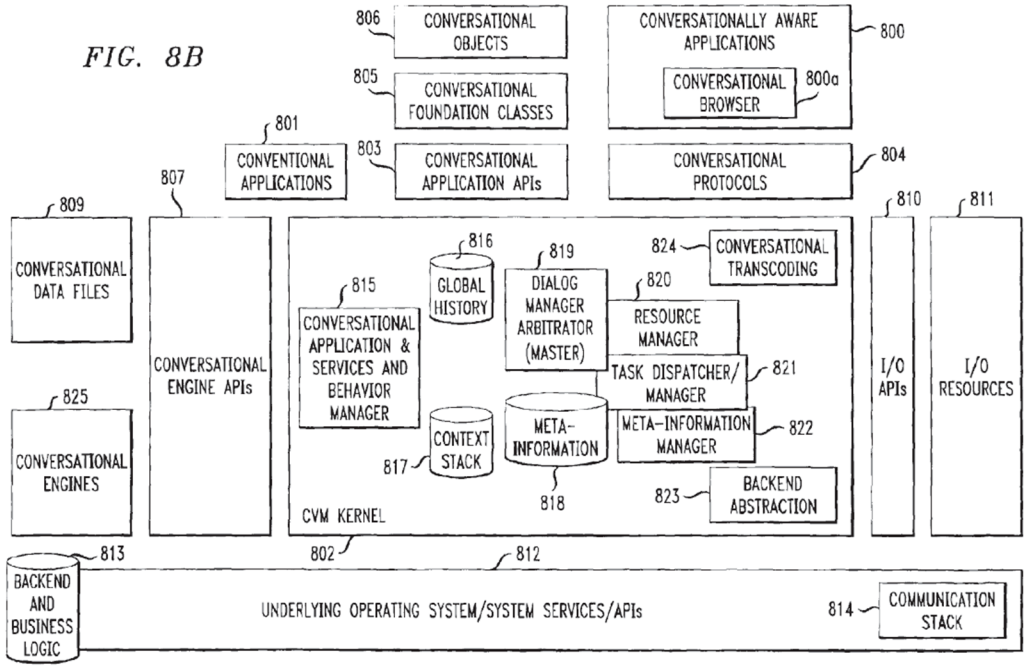

300
START
OBTAIN RAW MULTI-MODAL INPUT DATA FROM MULTI-MODAL DATA SOURCES
302
I/O MANAGER ABSTRACTS RAW INPUT DATA INTO EVENTS
304
ONE OR MORE RECOGNITION ENGINES PROCESS EVENT, IF NECESSARY, BASED ON THE NATURE OF THE INPUT DATA
306
308
STORE EVENTS ON CONTEXT STACK
MOOD/FOCUS CLASSIFIER DETECTS MOOD/FOCUS ASSOCIATED WITH USER
310
EXECUTE ACTION BASED ON DETECTED MOOD/FOCUS
312

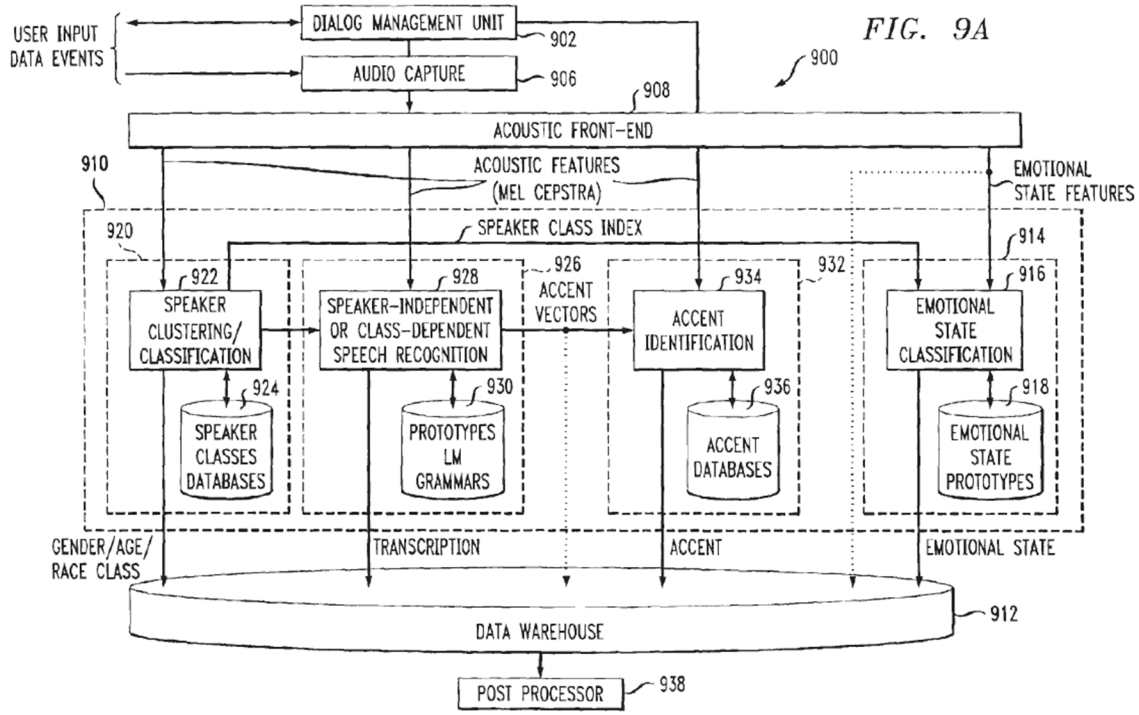
Claim 8 of the '957 Patent	Exemplary Citations to Maes
	Maes at 5D:  Maes at Fig. 6:

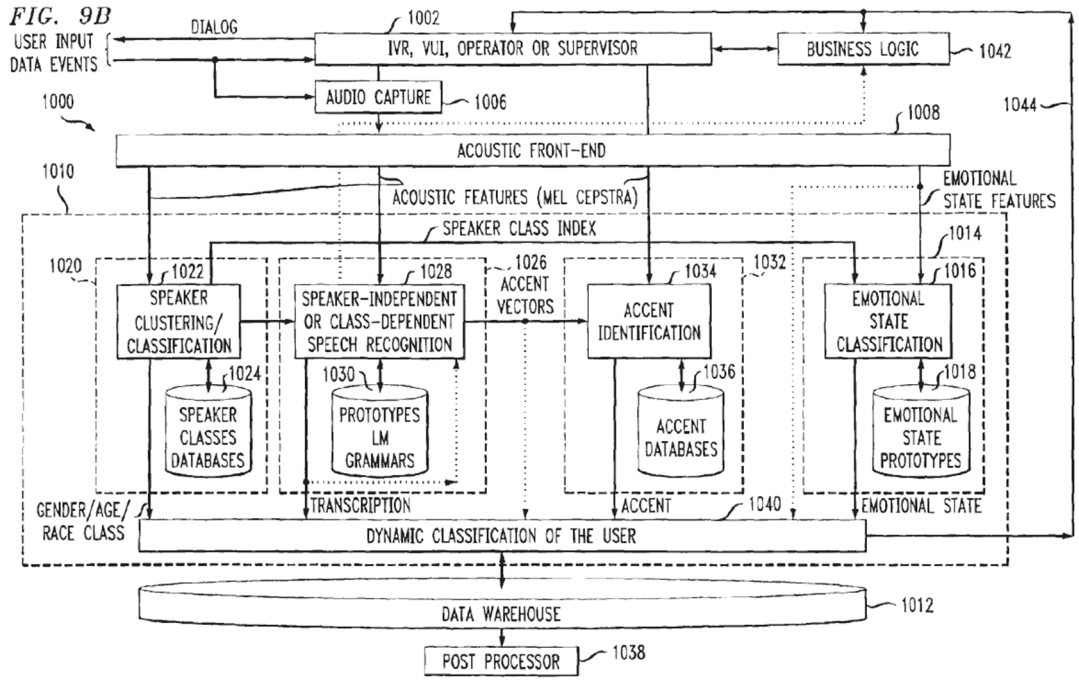
Claim 8 of the ’957 Patent	Exemplary Citations to Maes
	FIG. 6  Maes at Fig. 7:

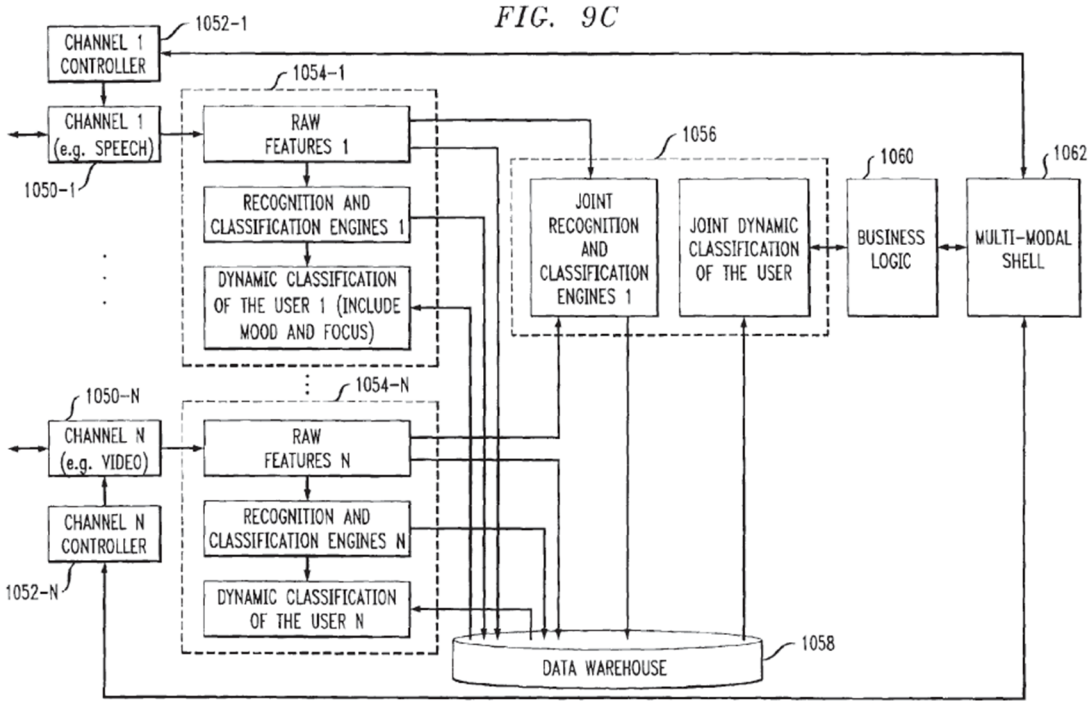
Claim 8 of the '957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 8A:

Claim 8 of the '957 Patent	Exemplary Citations to Maes
	 Maes at Fig 8B:

Claim 8 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 9A:

Claim 8 of the ’957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 9B:

Claim 8 of the '957 Patent	Exemplary Citations to Maes
	 Maes at Fig. 9C:

Claim 8 of the ’957 Patent	Exemplary Citations to Maes
	FIG. 9C  Maes at 7:60-8:65: Thus, the dialog manager must first determine the user's intent based on the current event and, if available, the historical information (e.g., past events) stored in the context stack. For instance, returning to the vehicle example, the user may say “turn it on,” while pointing at the vehicle radio. The dialog manager would therefore receive the results of the recognized events associated with the spoken utterance “turn it on” and the gesture of pointing to the radio. Based on these events, the dialog manager does a search of the existing applications, transactions or “dialogs,” or portions thereof, with which such an utterance and gesture could be associated. Accordingly, as shown in FIG. 1, the dialog manager 18 determines the appropriate CML-authored application 24. The

Claim 8 of the ’957 Patent	Exemplary Citations to Maes
	application may be stored on the system 10 or accessed (e.g., downloaded) from some remote location. If the dialog manager determines with some predetermined degree of confidence that the application it selects is the one which will effectuate the users desire, the dialog manager executes the next step of the multi-modal dialog (e.g., prompt or display for missing, ambiguous or confusing information, asks for confirmation or launches the execution of an action associated to a fully understood multi-modal request from the user) of that application based on the multi-modal input. That is, the dialog manager selects the appropriate device (e.g., radio) activation routine and instructs the I/O manager to output a command to activate the radio. The predetermined degree of confidence may be that at least two input parameters or variables of the application are satisfied or provided by the received events. Of course, depending on the application, other levels of confidence and algorithms may be established as, for example, described in K. A. Papineni, S. Roukos, R. T. Ward, “Free-flow dialog management using forms,” Proc. Eurospeech, Budapest, 1999; and K. Davies et al., “The IBM conversational telephony system for financial applications,” Proc. Eurospeech, Budapest, 1999, the disclosures of which are incorporated by reference herein. Consider the case where the user first says “turn it on,” and then a few seconds later points to the radio. The dialog manager would first try to determine user intent based solely on the “turn it on” command. However, since there are likely other devices in the vehicle that could be turned on, the system would likely not be able to determine with a sufficient degree of confidence what the user was referring to. However, this recognized spoken utterance event is stored on the context stack. Then, when the recognized gesture event (e.g., pointing to the radio) is received, the dialog manager takes this event and the previous spoken utterance event stored on the context stack and makes a determination that the user intended to have the radio turned on. Consider the case where the user says “turn it on,” but makes no gesture and provides no other utterance. In this case, assume that the dialog manager does not have enough input to determine the user intent (step 212 in FIG. 2) and thus implement the command. The dialog manager, in step 214, then causes the generation of an output to the user requesting further input data so that the user's intent can be disambiguated. This may be accomplished by the dialog manager instructing the I/O manager to have the I/O subsystem output a request for clarification. In one embodiment, the I/O subsystem 12 may comprise a text-to-speech (TTS) engine and one or more output

Claim 8 of the ’957 Patent	Exemplary Citations to Maes
	speakers. The dialog manager then generates a predetermined question such as “what device do you want to have turned on?” which the TTS engine converts to a synthesized utterance that is audibly output by the speaker to the user. The user, hearing the query, could then point to the radio or say “the radio” thereby providing the dialog manager with the additional input data to disambiguate his request. That is, with reference to FIG. 2, the system 10 obtains the raw input data, again in step 202, and the process 200 iterates based on the new data. Such iteration can continue as long as necessary for the dialog manager to determine the user's intent. Maes at 8:66-9:28: The dialog manager 18 may also seek confirmation in step 216 from the user in the same manner as the request for more information (step 214) before executing the processed event, dispatching a task and/or executing some other action in step 218 (e.g., causing the radio to be turned on). For example, the system may output “do you want the radio turned on?” To which the user may respond “yes.” The system then causes the radio to be turned on. Further, the dialog manager 18 may store information it generates and/or obtains during the processing of a current event on the context stack 20 for use in making resolution or other determinations at some later time. Of course, it is to be understood that the above example is a simple example of device ambiguity resolution. As mentioned, the system 10 can also make user ambiguity resolution determinations, e.g., in a multiple user environment, someone says “dial my office.” Given the explanation above, one of ordinary skill will appreciate how the system 10 could handle such a command in order to decide who among the multiple users made the request and then effectuate the order. Also, the output to the user to request further input may be made in any other number of ways and with any amount of interaction turns between the user and feedback from the system to the user. For example, the I/O subsystem 12 may include a GUI-based display whereby the request is made by the system in the form of a text message displayed on the screen of the display. One of ordinary skill in the art will appreciate many other output mechanisms for implementing the teachings herein.

Claim 8 of the ’957 Patent	Exemplary Citations to Maes
	Maes at 20:11-20: Then, a search is performed in search module 434 with language models (LM) based on the weighted probabilities received from module 430. That is, the acoustic units identified as having the highest probabilities of representing what was uttered in the arbitrary content video are put together to form words. The words are output by the search engine 434 as the decoded system output. A conventional search engine may be employed. This output is provided to the dialog manager 18 of FIG. 1 for use in disambiguating the user's intent, as described above. Maes at 33:26-34:11: Referring now to FIG. 8B, a diagram illustrates abstract programming layers of a CVM according to a preferred embodiment. The abstract layers of the CVM comprise conversationally aware applications 800 and conventional applications 801 that can run on top of the CVM. An application that relies on multi-modal disambiguation is an example of such a conversational application that executes on top of the CVM. Similarly, an application that exploits focus information or mood can be considered as a conversational application on top of the CVM. These applications are the programs that are executed by the system to provide the user with the interaction he desires within the environment in which the system is deployed. As discussed above, the conversationally aware applications 800 interact with a CVM kernel layer 802 via a conversational application API layer 803. The conversational application API layer 803 encompasses conversational programming languages/scripts and libraries (conversational foundation classes) to provide the various features offered by the CVM kernel 802. For example, the conversational programming languages/scripts provide the conversational APIs that allow an application developer to hook (or develop) conversationally aware applications 800. They also provide the conversational API layer 803, conversational protocols 804 and system calls that allow a developer to build the conversational features into an application to make it “conversationally aware.” The code implementing the applications, API calls and protocol calls includes interpreted and compiled scripts and programs, with library links, conversational logic engine call and conversational foundation classes.

Claim 8 of the ’957 Patent	Exemplary Citations to Maes
	More specifically, the conversational application API layer 803 comprises a plurality of conversational foundation classes 805 (or fundamental dialog components) which are provided to the application developer through library functions that may be used to build a CUI or conversationally aware applications 800. The conversational foundation classes 805 are the elementary components or conversational gestures (as described by T. V. Raman, in “Auditory User Interfaces, Toward The Speaking Computer,” Kluwer Academic Publishers, Boston 1997) that characterize any dialog, independently of the modality or combination of modalities (which can be implemented procedurally or declaratively). The conversational foundation classes 805 comprise CUI building blocks and conversational platform libraries, dialog modules and components, and dialog scripts and beans. The conversational foundation classes 805 may be compiled locally into conversational objects 806. More specifically, the conversational objects 805 (or dialog components) are compiled from the conversational foundation classes 805 (fundamental dialog components) by combining the different individual classes in a code calling these libraries through a programming language such as Java or C++. Maes at 34:34-58: Some examples of conversational gestures that may be implemented are as follows. A conversational gesture message is used by a machine to convey informational messages to the user. The gesture messages will typically be rendered as a displayed string or spoken prompt. Portions of the message to be spoken can be a function of the current state of the various applications/dialogs running on top of the CVM. A conversational gesture “select from set” is used to encapsulate dialogues where the user is expected to pick from a set of discrete choices. It encapsulates the prompt, the default selection, as well as the set of legal choices. Conversational gesture message “select from range” encapsulates dialogs where the user is allowed to pick a value from a continuous range of values. The gesture encapsulates the valid range, the current selection, and an informational prompt. In addition, conversational gesture input is used to obtain user input when the input constraints are more complex (or perhaps non-existent). The gesture encapsulates the user prompt, application-level semantics about the item of information being requested and possibly a predicate to test the validity of the input. As described above, however, the conversational foundation classes include, yet surpass, the concept of conversational gestures (i.e.,

Claim 8 of the ’957 Patent	Exemplary Citations to Maes
	they extend to the level of fundamental behavior and services as well as rules to perform conversational tasks). Maes at 36:8-48: The core CVM kernel layer 802 comprises programming layers such as a conversational application and behavior/service manager layer 815, a conversational dialog manager (arbitrator) layer 819, a conversational resource manager layer 820, a task/dispatcher manager 821 and a meta-information manager 822, which provide the core functions of the CVM layer 802. It is to be understood that these components may be implemented as part of the dialog manager 18 of FIG. 1. The conversational application and behavior/service manager layer 815 comprises functions for managing the conventional and conversationally aware applications 800 and 801. Such management functions include, for example, keeping track of which applications are registered (both local and network-distributed), what are the dialog interfaces (if any) of the applications, and what is the state of each application. In addition, the conversational application and services/behavior manager 815 initiates all the tasks associated with any specific service or behavior provided by the CVM system. The conversational services and behaviors are all the behaviors and features of a conversational UI that the user may expect to find in the applications and interactions, as well as the features that an application developer may expect to be able to access via APIs (without having to implement with the development of the application). Examples of the conversational services and behavior provided by the CVM kernel 802 include, but are not limited to, conversational categorization and meta-information, conversational object, resource and file management, conversational search, conversational selection, conversational customization, conversational security, conversational help, conversational prioritization, conversational resource management, output formatting and presentation, summarization, conversational delayed actions/agents/memorization, conversational logic, and coordinated interfaces and devices. Such services are provided through API calls via the conversational application API Layer 803. The conversational application and behavior/services manager 815 is responsible for executing all the different functions needed to adapt the UI to the capabilities and constraints of the device, application and/or user preferences.

Claim 8 of the ’957 Patent	Exemplary Citations to Maes
	Maes at 36:49-37:3: The conversational dialog manager 819 comprises functions for managing the dialog (conversational dialog comprising speech and other multi-modal I/O such as GUI keyboard, pointer, mouse, as well as video input, etc.) and arbitration (dialog manager arbitrator or DMA) across all registered applications. In particular, the conversational dialog manager 819 determines what information the user has, which inputs the user presents, and which application(s) should handle the user inputs. The DMA processes abstracted I/O events (abstracted by the I/O manager) using the context/history to understand the user intent. When an abstract event occurs, the DMA determines the target of the event and, if needed, seeks confirmation, disambiguation, correction, more details, etc., until the intent is unambiguous and fully determined. The DMA then launches the action associated to the user's query. The DMA function handles multi-modal I/O events to: (1) determine the target application or dialog (or portion of it); and (2) use past history and context to: (a) understand the intent of the user; (b) follow up with a dialog to disambiguate, complete, correct or confirm the understanding; (c) or, dispatch a task resulting from fall understanding of the intent of the user. Maes at 37:17-50: The task dispatcher/manager 821 dispatches and coordinates different tasks and processes that are spawned (by the user and machine) on local and networked conventional and conversational resources. The meta-information manager 822 manages the meta-information associated with the system via a meta-information repository 818. The meta-information manager 822 and repository 818 collect all the information typically assumed known in a conversational interaction but not available at the level of the current conversation. Examples are a-priori knowledge, cultural, educational assumptions and persistent information, past request, references, information about the user, the application, news, etc. It is typically the information that needs to be preserved and persist beyond the length/life of the conversational history/context and the information that is expected to be common knowledge for the conversation and, therefore, has never been defined during the current and possible past conversational interactions. Also, shortcuts to commands, resources and macros, etc., are managed by the meta-information manager 822 and stored in the

Claim 8 of the ’957 Patent	Exemplary Citations to Maes
	meta-information repository 818. In addition, the meta-information repository 818 includes a user-usage log based on user identity. It is to be appreciated that services such as conversational help and assistance, as well as some dialog prompts (introduction, questions, feedback, etc.) provided by the CVM system can be tailored based on the usage history of the user as stored in the meta-information repository 818 and associated with the application. If a user has been previously interacting with a given application, an explanation can be reduced assuming that it is familiar to the user. Similarly, if a user commits many errors, the explanations can be more complex, as multiple errors are interpreted as user uncertainty, unfamiliarity, or incomprehension/misunderstanding of the application or function. Maes at 38:3-17: The CVM kernel layer 802 further comprises a backend abstraction layer 823 which allows access to backend business logic 813 via the dialog manager 819 (rather than bypassing the dialog manager 819). This allows such accesses to be added to the context stack 817 and global history 816. For instance, the backend abstraction layer 823 can translate input and output to and from the dialog manager 819 to database queries. This layer 823 will convert standardized attribute value n-tuples into database queries and translate the result of such queries into tables or sets of attribute value n-tuples back to the dialog manager 819. In addition, a conversational transcoding layer 824 is provided to adapt the behavior, UI and dialog presented to the user based on the I/O and engine capabilities of the device which executes the CVM system. To the extent it is found that Maes does not expressly disclose this limitation, the limitation is inherent. And to the extent it is found that Maes does not anticipate this claim, Maes renders it obvious, standing alone, in combination with the knowledge of a person of ordinary skill in the art, and/or in combination with one or more other prior art references identified in the Cover Pleading or herein, for the reasons identified herein or in the Cover Pleading. A person of ordinary skill in the art would have been motivated to combine Maes with any of the other prior art references identified in the Cover Pleading or herein with a reasonable expectation of success because each of these references contains familiar elements within the similar fields of endeavor to address similar known problems, according to known methods which would yield predictable results.

Claim 8 of the ’957 Patent	Exemplary Citations to Maes
[b] receiving, in response to providing the prompt, a non-speech input, wherein the command or request is determined further based on the non-speech input	Maes discloses “receiving, in response to providing the prompt, a non-speech input, wherein the command or request is determined further based on the non-speech input[.]” *E.g.*, *see* Claim 7[b]. In addition, Maes at 7:60-8:65: Thus, the dialog manager must first determine the user's intent based on the current event and, if available, the historical information (e.g., past events) stored in the context stack. For instance, returning to the vehicle example, the user may say “turn it on,” while pointing at the vehicle radio. The dialog manager would therefore receive the results of the recognized events associated with the spoken utterance “turn it on” and the gesture of pointing to the radio. Based on these events, the dialog manager does a search of the existing applications, transactions or “dialogs,” or portions thereof, with which such an utterance and gesture could be associated. Accordingly, as shown in FIG. 1, the dialog manager 18 determines the appropriate CML-authored application 24. The application may be stored on the system 10 or accessed (e.g., downloaded) from some remote location. If the dialog manager determines with some predetermined degree of confidence that the application it selects is the one which will effectuate the users desire, the dialog manager executes the next step of the multi-modal dialog (e.g., prompt or display for missing, ambiguous or confusing information, asks for confirmation or launches the execution of an action associated to a fully understood multi-modal request from the user) of that application based on the multi-modal input. That is, the dialog manager selects the appropriate device (e.g., radio) activation routine and instructs the I/O manager to output a command to activate the radio. The predetermined degree of confidence may be that at least two input parameters or variables of the application are satisfied or provided by the received events. Of course, depending on the application, other levels of confidence and algorithms may be established as, for example, described in K. A. Papineni, S. Roukos, R. T. Ward, “Free-flow dialog management using forms,” Proc. Eurospeech, Budapest, 1999; and K. Davies et al., “The IBM conversational telephony system for financial applications,” Proc. Eurospeech, Budapest, 1999, the disclosures of which are incorporated by reference herein.

Claim 8 of the ’957 Patent	Exemplary Citations to Maes
	Consider the case where the user first says “turn it on,” and then a few seconds later points to the radio. The dialog manager would first try to determine user intent based solely on the “turn it on” command. However, since there are likely other devices in the vehicle that could be turned on, the system would likely not be able to determine with a sufficient degree of confidence what the user was referring to. However, this recognized spoken utterance event is stored on the context stack. Then, when the recognized gesture event (e.g., pointing to the radio) is received, the dialog manager takes this event and the previous spoken utterance event stored on the context stack and makes a determination that the user intended to have the radio turned on. Consider the case where the user says “turn it on,” but makes no gesture and provides no other utterance. In this case, assume that the dialog manager does not have enough input to determine the user intent (step 212 in FIG. 2) and thus implement the command. The dialog manager, in step 214, then causes the generation of an output to the user requesting further input data so that the user's intent can be disambiguated. This may be accomplished by the dialog manager instructing the I/O manager to have the I/O subsystem output a request for clarification. In one embodiment, the I/O subsystem 12 may comprise a text-to-speech (TTS) engine and one or more output speakers. The dialog manager then generates a predetermined question such as “what device do you want to have turned on?” which the TTS engine converts to a synthesized utterance that is audibly output by the speaker to the user. The user, hearing the query, could then point to the radio or say “the radio” thereby providing the dialog manager with the additional input data to disambiguate his request. That is, with reference to FIG. 2, the system 10 obtains the raw input data, again in step 202, and the process 200 iterates based on the new data. Such iteration can continue as long as necessary for the dialog manager to determine the user's intent. Maes at 8:66-9:28: The dialog manager 18 may also seek confirmation in step 216 from the user in the same manner as the request for more information (step 214) before executing the processed event, dispatching a task and/or executing some other action in step 218 (e.g., causing the radio to be turned on). For example, the system may output “do you want the radio turned on?” To which the user may respond “yes.” The system then causes the radio to be turned on. Further, the dialog manager 18

Claim 8 of the ’957 Patent	Exemplary Citations to Maes
	may store information it generates and/or obtains during the processing of a current event on the context stack 20 for use in making resolution or other determinations at some later time. Of course, it is to be understood that the above example is a simple example of device ambiguity resolution. As mentioned, the system 10 can also make user ambiguity resolution determinations, e.g., in a multiple user environment, someone says “dial my office.” Given the explanation above, one of ordinary skill will appreciate how the system 10 could handle such a command in order to decide who among the multiple users made the request and then effectuate the order. Also, the output to the user to request further input may be made in any other number of ways and with any amount of interaction turns between the user and feedback from the system to the user. For example, the I/O subsystem 12 may include a GUI-based display whereby the request is made by the system in the form of a text message displayed on the screen of the display. One of ordinary skill in the art will appreciate many other output mechanisms for implementing the teachings herein. To the extent it is found that Maes does not expressly disclose this limitation, the limitation is inherent. And to the extent it is found that Maes does not anticipate this claim, Maes renders it obvious, standing alone, in combination with the knowledge of a person of ordinary skill in the art, and/or in combination with one or more other prior art references identified in the Cover Pleading or herein, for the reasons identified herein or in the Cover Pleading. A person of ordinary skill in the art would have been motivated to combine Maes with any of the other prior art references identified in the Cover Pleading or herein with a reasonable expectation of success because each of these references contains familiar elements within the similar fields of endeavor to address similar known problems, according to known methods which would yield predictable results.