



# ASK Technology White-paper

Version: 1.3  
Date: 10 February 2010  
Author: Ludo Stellingwerff



Kralingseweg 229  
NL - 3062 CE Rotterdam

t +31 (0)10 225 0130  
f +31 (0)10 436 7007  
i [www.ask-cs.com](http://www.ask-cs.com)

## Change history

Version	Date	Summary of changes
1.0	7. December 2009	Initial version
1.1	20. December 2009	Added platform services
1.2	3. February 2010	Layout modifications
1.3	10. February 2010	Layout and textual changes

## Table of Contents

Introduction.....	3
Technological Concepts.....	4
Agent-based software engineering.....	4
State equilibrium and monitoring.....	4
Emergent behavior.....	5
Medium independence.....	6
Adaptive communication.....	6
Workflow management.....	6
Platform ASK.....	7
Components.....	7
Agents.....	7
ASK data model.....	8
Interfaces.....	8
Data synchronization.....	9
Runtime environment.....	9
User Interfaces.....	9
Web-based user interfaces.....	9
ASK Desktop end-user interface.....	9
Admin interface.....	9
Data visualization.....	10
GIS integration.....	10
IVR & Scripting.....	10



## Introduction

The ASK platform offers a unique adaptive communication system, aimed at accelerating human interactions. It stands out by providing large scale communication with unprecedented flexibility in personalization and exception handling. Each interaction through ASK, whether by phone, text or other medium, can offer personal messages, use personal preferences and can fully adapt to the personal agenda of the participants.

The design of ASK is a result of more than a decade of research into self-organization, and the role of software agents in socio-technical systems. This paper offers an overview of the concepts and technologies that have laid the foundation for the product and which have driven the design and implementation of ASK.

The product name “ASK” is an acronym for “Access Society's Knowledge”, which highlights an important aspect of ASK: through adaptive communication-enabled business process support, ASK provides access to the knowledge that lies within the existing societies. Examples of such societies could be: A corporate structure, a client pool, the general public (for Emergency Response tools), etc.

## Technological Concepts

### ***Agent-based software engineering***

ASK has been designed using concepts taken from the research realm of agent-based software engineering. This is a broad field of software research, especially aimed at providing effective distributed modularization, containment of responsibilities and a high-level modeling language.

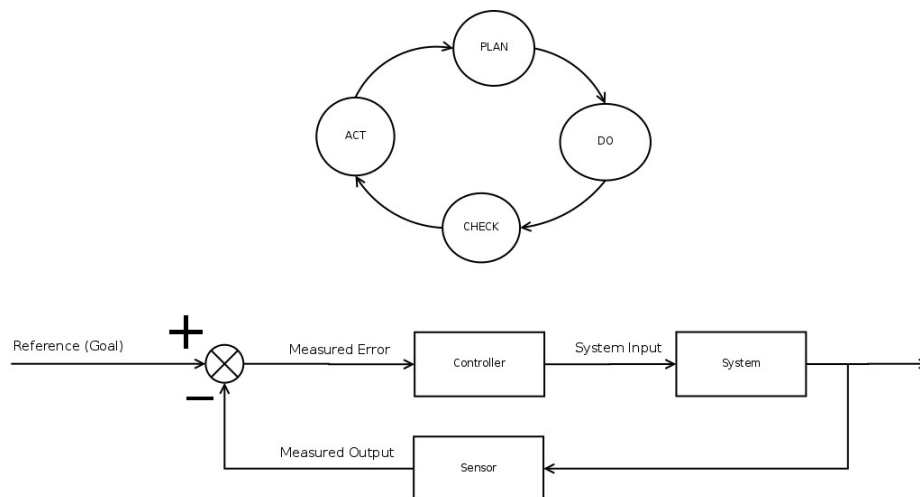
“Agent” is a broad term, used for various purposes and scale: On a high level it describes a separated entity with a certain level of autonomy. At implementation level it can be used for software modules, deployed programs and/or for actors within the data of the system.

Arguably the most important concept used in ASK is the agent's separated thread of control. Each agent is given a single task, with little to no shared data with other agents, which allows such an agent to operate independently and asynchronously from the other agents. This provides ASK a natural multi-threaded nature without the complex issues surrounding synchronization and locking. Because of the independence of tasks, ASK is highly resilient against escalation of local failures. If some process within ASK fails, the system as a whole will remain largely non-affected.

The ASK system has very little singletons: a given task is not dependent on one agent alone. If such an agent fails, other agents can take the task and finish the required work. Due to the way the agents monitor the available workload, this provides a high level of self-repair.

### ***State equilibrium and monitoring***

The research field around control systems has had an important influence on the ASK design. Control systems engineering is the engineering field that works towards maintaining a dynamic equilibrium within a system. Examples of such systems are the flight controls of an airplane, the central heater system of a house, the management Plan-Do-Check-Act cycle and a human keeping his balance, etc. One requirement for a dynamic equilibrium is to have a feedback mechanism about the current state of the system, which can be used to control the future state. Illustration 1 gives a standard presentation of such feedback systems, as used in control systems engineering and management.



*Illustration 1: Feedback system models*

ASK has a proactive monitoring agent, called the Scheduler, which is responsible for checking the current state of a given part of the system. If something happens in the outside world that influences this state, through feedback and inbound events, the Scheduler can check if this state is still within the required demand. If the deviation from the demanded state is too large, the Scheduler initiates actions to mitigate this problem. These actions can be directly aimed at mitigation or can be an escalation to a higher level within the system, which will be able to provide mitigating actions. Illustration 2 provides a diagram of the feedback system as implemented within ASK.

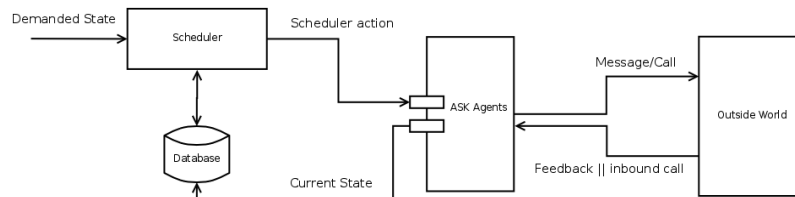


Illustration 2: ASK's feedback system

## Emergent behavior

The two concepts mentioned (agents and state maintenance) lead to emergent behavior of the system as a whole. Emergence is defined as: “Arising or occurring unexpectedly or unintentionally”, in this context it refers to the structures that become apparent within chaos, patterns that emerge from many (seemingly) independent events. Or as Jeffrey Goldstein defines it: “the arising of novel and coherent structures, patterns and properties during the process of self-organization in complex systems”.

Below are two analogies to illustrate this concept:

*In some way the ASK agents behave like ants. In the real world, ants are able to establish trails to find the shortest path from their nest to a food source. They achieve this not by direct communication but by making small local modifications to their environment by laying down "pheromones". The technology uses algorithms based on this behavior as an optimization technique.*

The ASK product is similar to an Ant colony. By using many agents, each working on the system state in parallel (asynchronously), the system as a whole exhibits emergent behavior. This is very much like the way human organizations function, for example in traffic, crowds, trade, etc.

*ASK is also like a soccer team: Each player has its own task, effecting only the local area around the player. But through teamwork and continuous interaction between the players, the game as a whole has several emergent aspects: Goals are scored, one teams wins and the crowd is amused (and a lot of money has changed hands).*

The ASK agents form a team, working on a common goal. If one or more agents get a "red card" and fail to deliver, other agents will pickup the task to achieve the goal.

---

## **Medium independence**



ASK offers a true multi-channel solution for communication: In its design, ASK can use any available communication medium for which its practical to create an interface.

In its design model, this medium could be smoke signals, but building an interface to such a medium is largely irrelevant. The media for which ASK does currently have interfaces are: Telephony (VoIP and ISDN), SMS, Email and Twitter.

Although ASK is largely a communication media independent platform, different media offer a different timing context. A single instance of ASK is capable of several thousands of transactions per minute, but in most situations there is a limited number of phonelines available. Compare this with Email or SMS, in which case this amount of transactions does become an important number. The actual bottleneck of the system therefore depends on the choice of media and scenario.

## **Adaptive communication**

Adaptive communication is the “Raison d'être” of the ASK product. All communication between people can be seen as one-on-one transactions between peers. ASK is designed to provide coordination of these transactions. This coordination is done through various tools, working together to reach the optimal communication scenario. One important optimization choice that ASK can make, is the choice of medium.

Various information sources can be used in the coordination algorithms, collected by ASK. These sources include meta information of the ongoing transaction, like phone numbers, user phone input (DTMF) sequences, email fields, etc. But they also include data from other customer information systems and webservices. The coordination is adaptive, in the sense that ASK can learn from earlier transactions to determine the best solution for future transactions.

An important aspect of the way ASK coordinates transactions, is time. ASK is capable of using delays, schedules and agenda's in the determination of the optimal transaction flow. Due to the massive multi-threading nature of the ASK agents, ASK can prioritize selected messages, guaranteeing on-time delivery. This is achieved in a near lock free manner, making the product extremely scalable, distributable and flexible.

## **Workflow management**

One important aspect of IT is the ability of software to support the workflow of a company. Many dedicated workflow solutions exist, many integrated within domain specific tools. ASK offers unique capabilities for workflow support, because of the communication aspects. All transitions within the workflow can be monitored in ASK, with extensive escalation and data driven decision support features to provide an optimal solution.

## Platform ASK

### Components

A given ASK installation consists of a number of components, at the top level four components can be determined:

- GUI
- Database
- Webservices
- ASK engine

In the near future these four will be joined by two utility tools:

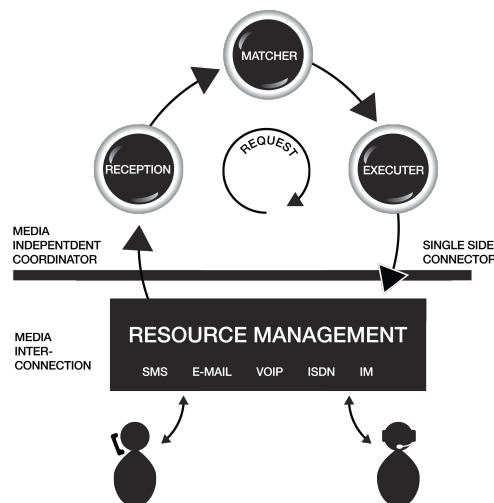
- IVO (Database synchronization tool, “Input versus Output”)
- AdaptTime (Agenda synchronization tool)

### Agents

The ASK engine consists of a large group of software agents, bundled into four types:

- Reception agents
- Matcher agents
- Resource management agents
- Scheduler agents

In most setups these agents are bundled into separate executables, which results in names like “The Reception”. But it is important to keep in mind that they are actual groups of independent agents, each agent with a single task to fulfill. The reception, matcher and resource manager agents work together to form a 'request' loop, see illustration 3.



*Illustration 3: Coordination loop*

The resource manager organizes the available media, and provides a real-time connection between the various participants in a conversation. The reception and matcher agents provide the decision support for managing the workflow of the conversation. The reception provides scripting and identification capabilities; the matcher provides selection, filtering and sorting services.

The scheduler agents are an important part of the pro-active nature of ASK. There are two important types of scheduler agents: Those that monitor a given demand and those that monitor and control business rules (hence their name: “business rulers”). If a certain demand is not met, the scheduler agent act like a person, calling through the system to reach users (agents or people) with the purpose to mitigate the problem.

## ASK data model

The data model of ASK is aimed at modeling the relevant state of the outside world. The core concept within this model is again an 'agent'. Each known person is represented by one or more agents, in the form of users. These users can be classified in groups. The third element in the model is the user's planboard, an agenda showing the reachability and availability of the user for communication. The user/agent model can represent several entities: actual human users, but also virtual users like tasks, organizational units, locations, etc. Through the use of the planboards, each of these users can be given a state, which can be monitored during the passage of time. When the system (through the Scheduler) detects that one or more agents fail to adhere to a required demand, it can start communication transactions (conversations) to mitigate the problem.

## Interfaces

ASK can be integrated within existing IT infrastructures through a variety of interfaces. These include webservices, database synchronization and communication channels. ASK's agent design provides the required tooling to let these interfaces remain sufficiently stable. Depending on the requirements of a specific market domain, a translation layer can be added based on the ASK interfaces.

For example, it is possible to add ER specific interfaces like the 'Common Alerting Protocol' CAP on top of the data model of ASK.

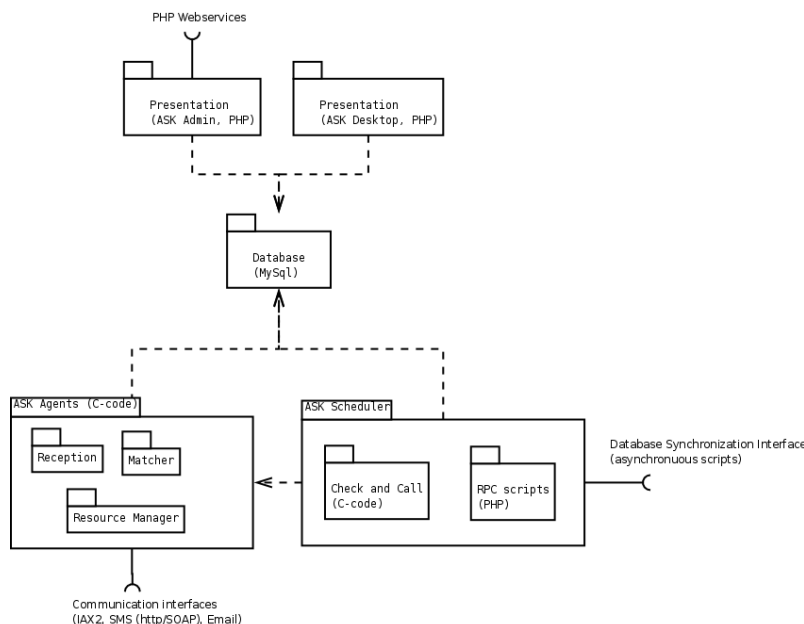


Illustration 4: Deployment and interfaces diagram

---

## **Data synchronization**



One important interface to the existing IT infrastructure is the data synchronization tooling ASK has available. Either through the use of separated tooling like IVO (data sync tool) or through specific ASK scheduler jobs data can be exchanged with various external sources. This can either be done through a dedicated protocol, XML-format or through the use of a datamart database environment. The latter is a proxy database which both involved partners (ASK and the domain specific database, e.g. SAP) keep synchronized. By using a proxy database local database structure changes won't force the other party to change their synchronization scripts.

## **Runtime environment**

For ease of deployment, maintenance, and support, ASK usually runs within a VMware virtual server. This offers a standardized environment, high-availability and hardware independence. Through cooperation with our hosting partner Luna, ASK currently runs in a top-level blade environment, with full redundancy on hardware and network facilities.

The product is implemented and tested to work on a Debian Linux OS, but portability to other platforms remains a good option, as care has been taken to only use standardized tooling and (UNIX) services. ASK consists of code written in C(++), Java, PHP, JavaScript and Flash(ActionScript).

## **User Interfaces**

### **Web-based user interfaces**

The Graphical User Interface (GUI) of ASK comes in two flavors: The Admin interface and the ASK Desktop.

### **ASK Desktop end-user interface**

The desktop is an web-based environment, in which several ASK features have been made accessible for end-users. It contains a highly adaptive environment in which users can control large scale communication scenarios:

- Inquiry and classification (asking questions and grouping people according to the given answers)
- Notification (passing a message to a large group of people (e.g. Emergency response market))
- Pool management (monitoring and controlling the minimum availability of a group of people for a given job)

### **Admin interface**

The admin interface offers a viewport into the data models used within ASK. At this inner level it offers a toolset for monitoring and controlling the way ASK handles its tasks and scenarios. This environment is only aimed at administrators and system maintainers, as it requires a good level of understanding of the inner workings of ASK and its way of modeling the actual application domain.

## ***Data visualization***

One important aspect of the ASK data model is the agenda of the users. ASK has a visualization of this agenda information for getting an effective overview of the time aspects of the user model. An image says more than a thousand words, especially if those words are provided in the form of a large data table.

Similarly it is important to provide an overview of the groups in the system. Through the use of a group cloud visualization ASK offers a quick click through access to all groups and users.

## ***GIS integration***

In several scenarios there is a geographical location dependency for the communication decisions that ASK is making. For example in track and tracing, planning and emergency response, the location of a given event and/or person is important. Through the use and integration of geographical information systems (GIS), e.g. Google Maps, ASK offers a visualization of this information, including tools to effectuate communication from these maps.

## ***IVR & Scripting***

For most communication scenarios ASK will require a plan or script. ASK provides a scripting language for control of its communication flow. In the context of phone conversations this script acts like an interactive voice response (IVR) language.

In the ASK implementation, these scripts are highly dynamic, largely due to their data driven character. But this poses a challenge: Providing such a script requires programming skills. To alleviate and mitigate this issue, ASK contains a web-based graphical scripting environment, in which system maintainers can create the scripts in a drag-and-drop fashion.