1  
**Deliverable**

**“A2A3D – Awareness handling infrastructure: position and specification”**

**FP7 Capacities Programme:** Research Infrastructures

**Project title:** VISION Advanced Infrastructure For Research

**Acronym:** VISIONAIR

**Grant agreement n°:** 262044 – Integrating Activity (Combination of CP-CSA)

**Project starting date:** 2011/02/01

**Project duration:** 48 months

**Summary and scope of the document:**

This document presents the specification of the awareness handling environment proposed in WP9 Action 13 and in WP 11.

Due date of deliverable:

Actual submission date: M12

Work Package: 9, 11

Version: 15/09/2014

Dissemination level: **PP** (restricted to other programme participants including the Commission SERVICES).

**Lead beneficiary for this deliverable: SZTAKI (partner 21)**

**Author(s): Zoltan Toth**

**LIST OF CONTRIBUTORS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Partner** | **Acronym** | **Laboratory Name** | **Name of the contact** |
| SZTAKI | SZTAKI | SZTAKI DSD | Laszlo Kovacs |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**TABLE OF CONTENTS**

[Introduction 4](#_Toc404589997)

[Participants to WP9 - Action 13: Action to awareness in 3D environments 4](#_Toc404589998)

[Vision 4](#_Toc404589999)

[Relation to tasks in WP 9 and 11 4](#_Toc404590000)

[Task 9.2: Modalities of interoperability between remote devices 4](#_Toc404590001)

[Task 11.3: Research visualization-related ergonomics, usage, and social impact 5](#_Toc404590002)

[A2A3D requirements 6](#_Toc404590003)

[A2A3D specification 7](#_Toc404590004)

[Event 7](#_Toc404590005)

[EventSource 9](#_Toc404590006)

[Event source repository 10](#_Toc404590007)

[Static configuration 10](#_Toc404590008)

[Dynamic configuration interface 10](#_Toc404590009)

[Visualization web interface 10](#_Toc404590010)

[Security 11](#_Toc404590011)

[Reliability 11](#_Toc404590012)

[Performance 11](#_Toc404590013)

# Introduction

Awareness provides a way to get informed or understand the situation regarding entities of interest. Awareness information generation deals with deciding what context and raw information is necessary to create awareness of a specific situation or state. For example, to generate data that describe the fact that a specific user is “actively working“ requires not only the “user presence” context information but also historical data about what he has done. One important research topic in this area is to figure out what information and in what details is necessary or useful to be presented for the users. The importance of awareness information increases in the case of remote collaboration, where the better understanding of the remote partners’ status might be essential for the efficient collaboration.

SZTAKI proposed a generic awareness handling system (Action to Awareness in 3D Environments – A2A3D) which is suitable to support of the needs of Visionair environments. This document positions this approach within the Visionair goals and gives the specification of the environment.

# Participants to WP9 - Action 13: Action to awareness in 3D environments

|  |  |  |
| --- | --- | --- |
| **Short Name** | **Partner Number** | **Pers-months** |
| USTUTT | 6 | 4 |
| TECHNION | 12 |  |
| SZTAKI | 21 | 4,80 + 3,20 |

# Vision

MTA SZTAKI proposes a generic awareness handling system suitable for Virtual environments. This proposed system consists of a central message repository and distribution server, several data collector agents and several data consumer agents. The data collector agents collect all available and feasible event data from the collaboration environment and publish them to the repository server, while the consumers subscribe to these event notifications and present them to remote users and services. This setup facilitates to enrich the collaboration of remote teams with awareness data and via using the persistent repository, it provides a log of activities for both local and remote users.

# Relation to tasks in WP 9 and 11

The proposed awareness handling system mainly contributes to the Visionair WP9 (Advanced methods for interaction and collaboration) and WP11 (Visualization of conceptual models and conceptual modelling). In this chapter we define the relation of this system to the given tasks and subtasks in these work packages.

## Task 9.2: Modalities of interoperability between remote devices

This task focuses on the environmental conditions of integration of methods and metaphors.

The results of this task will be applied to WP6 on Virtual Reality and WP7 on collaborative environments by proposing synchronous and asynchronous collaboration between different platforms. Specifically, we propose three fundamental sub-tasks to tackle different points.

### Sub Task 9.2.1: Getting independent of the platforms

The first one, from a local point of view, will aim at defining temporal model of 3D user interfaces that can be used to normalize the integration way of common or novel 3D user interface metaphors in a more robust manner and unified way. To this purpose, the definition of a clear API that separates the modelling world from the platform will lead to a definition of portable code. The defined scalable “kernel” for 3D interaction should be embedded into many VR or visualisation pipelines and should be completed by the definition of a toolbox of code for migrating desktop applications to immersive applications. The condition of distribution ensuring efficient collaboration on distributed workspaces with different softwares will lead to software development.

A2A3D contributes to this sub task with a framework, where local, installation dependent data collector agents might communicate with other agents which might be located in a remote installation. If meaningful event data might be collected on the data collector side, it gives a platform independent way of event information sharing.

### Sub Task 9.2.2: Collaborating locally

The second will address the question of co-located workspace. This is very important for design environments (WP7) and for immersive environments (WP6) and should rely on environments proposed by the partners. Driven through realistic case studies from engineering, we would explore the underlying collaboration platforms and interfaces that are necessary for co-located teams to solve effectively complex design problems. More, driven through typical immersive applications where co-located users have to collaborate on the same virtual environment, we will rely on the developments proposed within task 9.1 and Task 9.3 to adapt them to a multi-user use. Besides this co-located collaboration, offline or co-temporal collaboration becomes more and more important. Here, work partners do not need to specify when to meet and discuss their datasets. Instead, they can work independently from each other and share their insights later. As such, in co-temporal collaboration, where actions are not only geographically distributed, asynchronous work contexts are becoming an important issue. As a consequence, an annotation system suitable for virtual environments and easily adaptable to different display technologies and interaction metaphors must be developed.

The proposed persistent repository provides a log of activities which might support the local collaboration.

### Sub Task 9.2.3: Collaborating distantly

Third, the addressed goal will be distributed workspaces. The focus will be to explore the methods and tools that are necessary for remote teams to work together as if they are in the same co-located space but with different platforms. The services that are necessary to offer future collaborative online or offline problem solving environments may rely on the use of new software and on collaboration models.

Remote collaborations are supported with immediate awareness information by the proposed WP 9 and 11 action.

## Task 11.3: Research visualization-related ergonomics, usage, and social impact

In this task we shall first define virtual environments enhanced social setting, including user experience and expectations, collaborative tasks and their performance, and methods for evaluating how good a particular virtual environment is for supporting collaboration. Using various settings from scientific visualisation for exploration, engineering, building construction, and urban planning, extract generic and domain-specific features to be supported within typical collaborative environments. Based on features identified, we shall develop a configurable model-driven visualization-based social framework that supports various application domains, environments, user profiles, and usage scenarios via a combination of appropriate technologies and interfaces. We shall model usage scenarios and social impact of multi-functional teams and reconciliation of different objectives in a collaborative session. We shall develop a dynamic and reconfigurable social environment to support both individual and team perspectives, and explore barriers for interaction for teams involved in a collaborative session using heterogeneous VR displays, possibly including avatars, CAVE, Powerwall, and Surface Tables, and suggest means and/or novel technologies to overcome such barriers. An application synchronisation and steering based on CoSpaces to complete virtual collaborative working environment and a unified model driven user interface for virtual environments, desktops and other devices will be developed.

# A2A3D requirements

The development of A2A3D is a research activity, with a given scientific and practical goals. The main result should be a complete and working awareness handling framework where based on the defined use-cases of the project, different scientific evaluations might be executed. The long term goal is to support collaboration in 3D environments with awareness information.

Since all Visionair installations are different and consist of unique software tools and hardware equipments, A2A3D will be inserted into a quite heterogeneous environment. In this heterogeneous environment it has to work as a bridge between the entities of the system and the users of the system. From higher level of perspective we can say, it collects data from the Visionair system entities, processes, stores, transfers and visualizes this data.

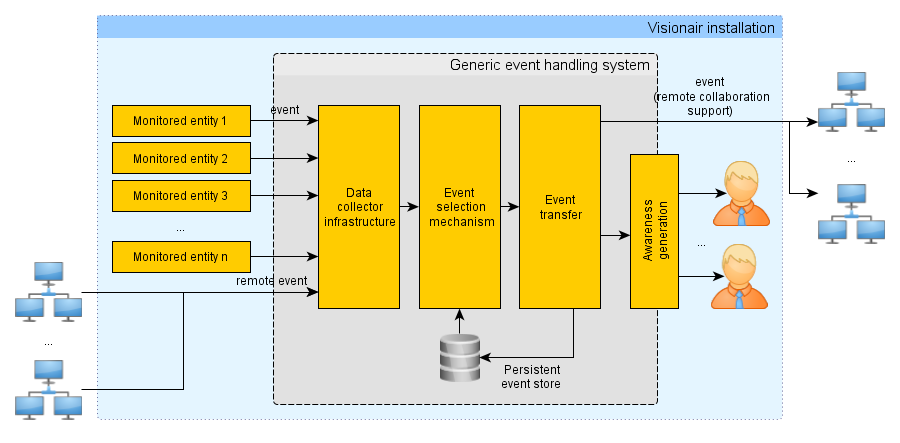


Figure 1: Event to Awareness lifecycle

From implementation point of view this environment results systematic processing of events from various sources (Figure 1). Thus the system must be capable of:

* Collecting events from various sources;
* Processing events and generating awareness information;
* Transfer the awareness information (to support remote collaboration);
* Visualize awareness information (own interface or already existing visualization interfaces);
* Collect and visualize remote awareness information;
* Store awareness information for later evaluation.

The generic use of the A2A3D infrastructure adds the following requirements to the system:

* The data collection must be a flexible, pluggable infrastructure;
* The event to awareness processing unit must be a flexible, pluggable infrastructure.

This might result a network of several plugged in data collector and event processing entities in the system, which must use the system resources efficiently to reduce the computational overhead.

# A2A3D specification

Based on the requirements above we propose a system consisting of individual, identical agents (Figure 2). The intended implementation platform is JAVA. Since this is the specification of a beta prototype, special security issues might be mentioned here but will not be handled in the first implementations.

The system as a whole consists of at least one A2A3D agent, or a distributed set of A2A3D agents. This/these agents are integrated in their environment, harvest data from them and notifies the users about events. A single A2A3D agent is an autonomous entity. The agent itself is a network of EventSource items, a database, a configuration connecting the EventSources and describing the behaviour of the whole system, and a web interface which might be used to establish connections to other agents. The main purpose of the system is generating and handling of Event objects.

## Event

Event is a software message indicating that something has happened. In our case this means monitoring entities about the state and state changes of the monitored system. The properties of the Event instances are:

#### date(Date)

The timestamp of the event. It is a local timestamp of the source system.

#### sourceMachineId(String)

Since the agents are working in a highly distributed system it is essential to be able to identify the proper source of a given event. Unfortunately, it is difficult to find a good URL (readability, uniqueness, simplicity). As a simplification, we will use the identifier set in the configuration file of the installation. This way different agents might be associated with same sourceMachineId which arises security issues (an agent might behave as another one), which should be handled later (e.g. choosing the MAC address of the system running the agent, but would go against the readability).

#### eventType(String)

The type of the event. The different event sources and data collectors are responsible to type the events. Based on the type, the event consumers might decide if they can handle them, and what metadata is wrapped in the content. During the development the ontology of event-types must be decided. (E.g. READ, WRITE, MOVED, UPDATED, etc.)

#### localObjectId(String)

The source object of the event. Together with the sourceMachineId this uniquely identifies the object of the event. (E.g. In the case of files it might be the file path.)

#### content(String)

The content of the event. It is a string containing serialized key-value pairs. It might refine the information already stored by the eventType. (E.g. for a COPY event it could store what was copied from which opened document.)

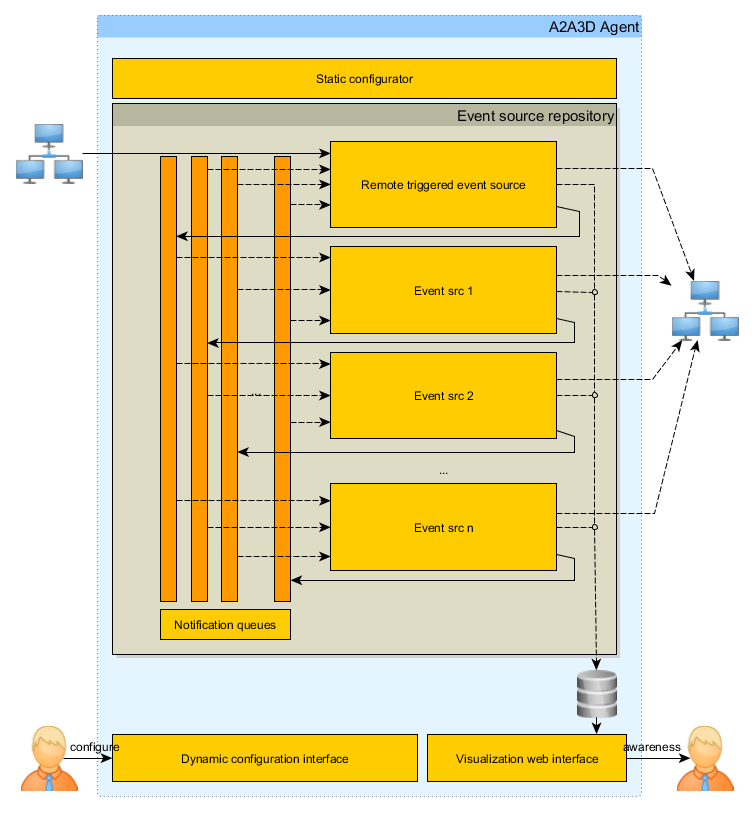


Figure 2: A2A3D schematic view

## EventSource

The EventSources are responsible for collecting, generating and forwarding the Event objects in the A2A3D agents, and generating notifications/awareness based on the Events. Each has an own configuration and glued together via notification queues. When an EventSource generates an Event, it passes it to the other EventSources subscribed for its notification queue. The data collection mechanisms are programmed in the EventSource implementations.

The remote communication between the A2A3D agents is also implemented via dedicated EventSource items.

The EventSources might have a filtering mechanism applied for the incoming Events, where different filters matched to the event descriptors. Only the matching events are processed by the given EventSources.

### Predefined event source elements

#### DbWriteEventSource

Stores the incoming events to a persistent DB store.

#### HttpPostEventRouter

Packs and sends the events to remote consumers (e.g. other A2A3D agents) via HTTP Post requests. It does not change the attributes (type, date, content) of the Event.

#### LogTailEventSource

Implementation of the UNIX tail -f command. Changes in the monitored file generates an Event object.

#### LoggerEventSource

Writes event descriptors into a specified log file. It is mainly for debugging the system.

#### QuartzSimpleTimedEventSource

Triggers events in given time periods. Might be used to initialize measures executed by other EventSources.

#### RemoteTriggeredEventSource

A simple web interface. It makes possible the communication between the agents. Remote agents get event notifications via this interface from other ones. It might also be used to request monitoring data from an agent.

#### UserAlertEventSource

A simple visualization interface. Shows an alert window with the details of the incoming event.

## Event source repository

The collection of the EventSource units in the agent.

## Static configuration

While the constitution of the agents are identical, they might behave completely different based on their configuration. The static configuration resolves the static settings of the system. Beside the usual database and port setting tasks, it sets the parameters of the EventSources and specifies the connections between them, too.

## Dynamic configuration interface

The event routers might be configured dynamically through a web interface. This mechanism makes possible to implement subscription for events by remote hosts. The configuration might happen via specific service messages.

## Visualization web interface

An interface to the persistent store to list events stored by the event db writer.

# Security

The first version of the implementation will neglect security questions. Later it is a must, to use secure connection in the agent communication, authentication of agents between themselves and authentication in the administrator interface.

# Reliability

The first version of the implementation will neglect reliability. If the use cases require more strict handling of this question than the optimistic message sending, then a handshaking protocol must be introduced into the system, and a method to store the unsent events permanently in persistent databases.

# Performance

The first version of the implementation will neglect the performance related questions.