

## Functional architecture of the SAPI framework

### Survey

This document summarises the content of the deliverable “**D1.3\_2 Functional and user requirements of the SAPI framework**” in relation to *RI 1.3 Framework requirements and architecture for the customised delivery of content and services*, within the scope of the first Development Objective (DO 1) “Study and analysis of the state of the art”.

Definition of the framework's functional architecture plays an important role in developing the platform insofar as it lays the base for embarking on research activities aimed at selecting suitable implementational technologies.

The study conducted herein precedes activities to analyse the technological state of the art and the characteristics and requirements of potential platform users, and represents the first stage in design of the SAPI framework (as shown in Figure 1).

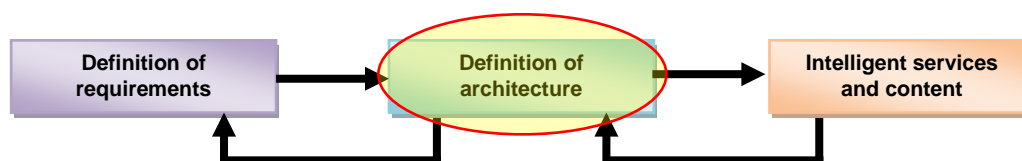


Figure 1: SAPI design

The activities carried out define the key characteristics of the functional architecture of SAPI, designed to allow the supply of intelligent services, i.e. services that can be customised and adapted to the requirements of the user profile and context. The activities are completed with a detailed description of the modules comprising the macro-blocks, specifying the specific functions and input/output behaviour for each of these.

The SAPI framework (Figure 2) makes available a series of functions aimed at allowing customisation of man-machine interaction within the context of e-business, with specific (but not exclusive) reference to the supply of online services and content and accessibility support for interaction with visually-impaired individuals.

SAPI belongs to the category of *Multichannel and Multimodal Context-Aware Customisable Information Systems*; hence by its very nature, its behaviour at a given moment and for a given user is heavily dependent on the *user* profile and *context* it finds itself in. Specifically, the designed system adapts its behaviour in relation to the user-context profile, in other words to the objective and subjective characteristics of the user, the context of use (environment, terminal, special peripheral networks) as well as the current interaction status and past history.

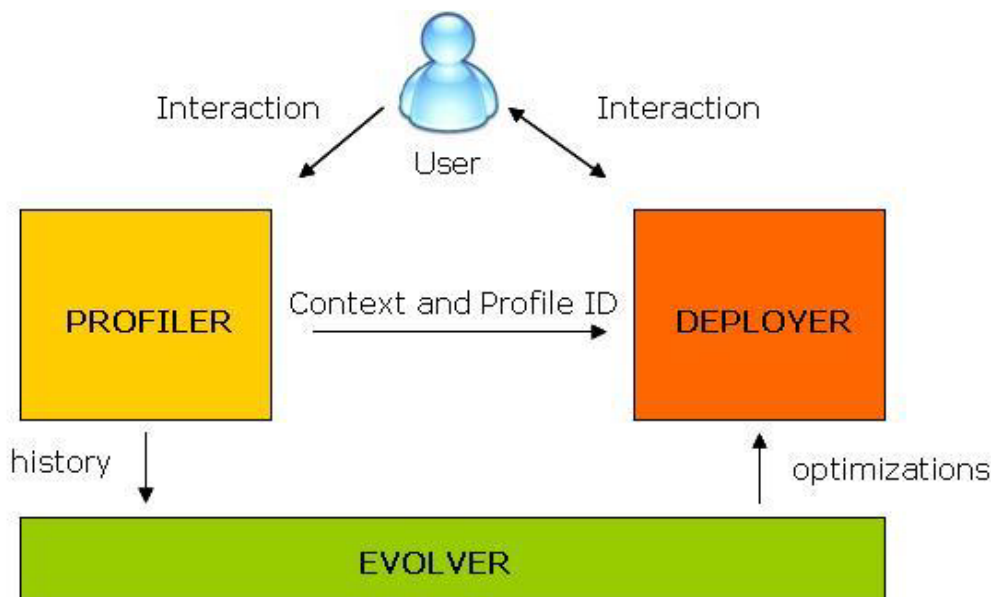


Figure 2: SAPI Framework

As shown in the figure, the SAPI framework is split into three functional macro-blocks called Profiler, Evolver and Deployer, which interact with each other and the user in order to ensure the most customised use of services possible.

After a preliminary user identification phase, the functional block known as Profiler deals with identification of the user-context profile to be associated to the specific user-ID. This is an all-important phase for allowing correct customisation of the service to be provided. Profiler also deals with saving the information captured during system use in order to update the profile history.

The latter will form the input of the functional block called Evolver which deals with optimising the service adaptation logic and evolution of the user and context profiling logic in order to ensure a high level of quality of system interaction with all potential users.

Lastly, Deployer is responsible for managing service composition and provision in response to user requirements. This block makes use of the optimisation information produced by Evolver and information regarding the user and context supplied by Profiler to compose the service in such a way as to ensure the accessibility, usability, adaptability and adaptivity requirements that are all-important in order to provide customised services.

Said architecture is designed to support the supply of intelligent services and content, in other words services able to self-adapt to the user and context.

Figure 3 offers an overall view of the functional architecture of the SAPI framework in which the functional set-up of each macro-block and input/output links between them are highlighted.

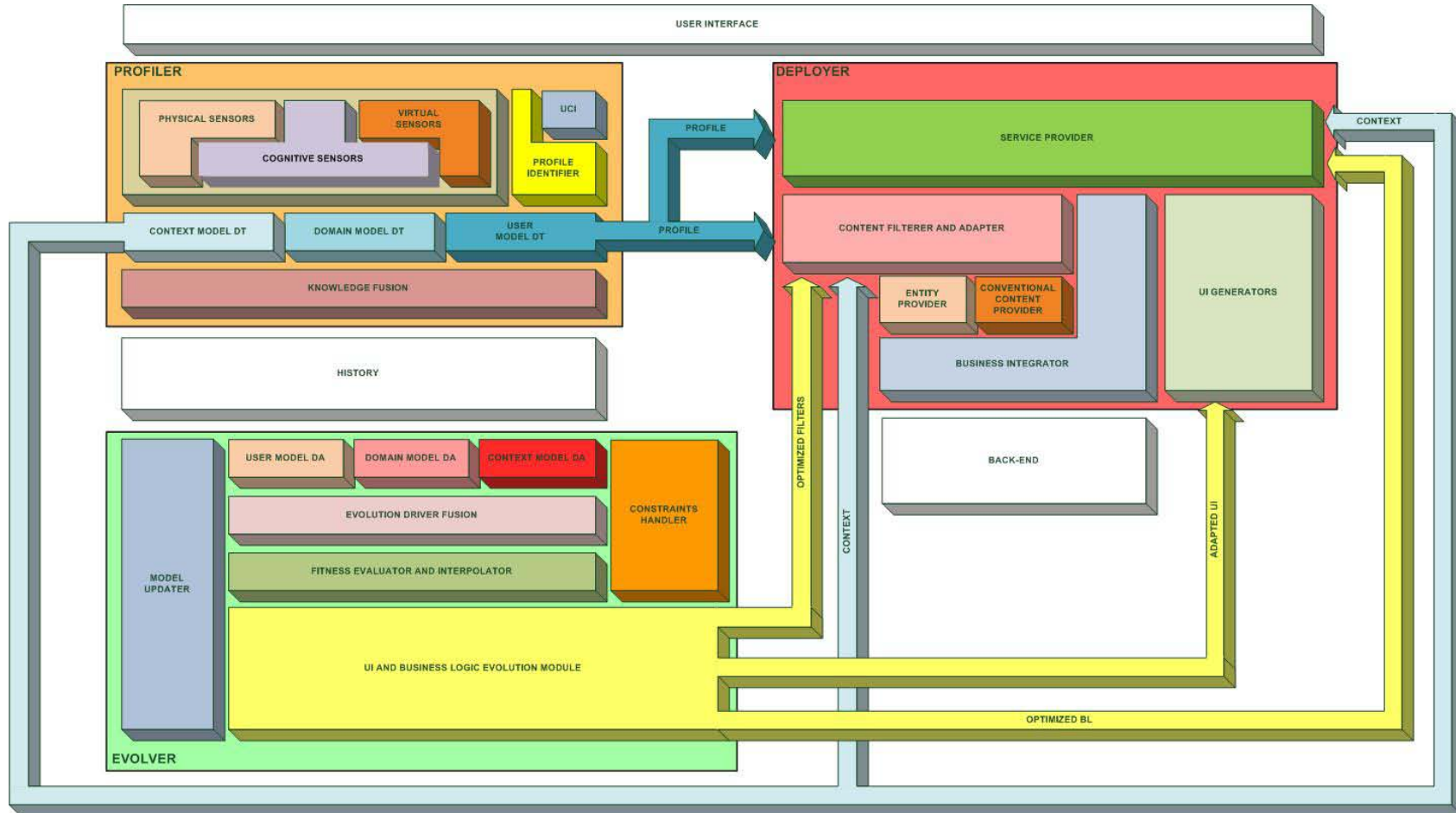


Figure 3: Functional architecture of the SAPI framework

Specifically, Profiler features a series of sensors responsible for acquiring data related to the user interaction session which serve to define the user context profile. Said data represent the input of a series of software modules (Data Transformers and Knowledge Fusion) which, on the basis of user interference rules and models, domain and context, reprocess and transform raw data into structured information which is subsequently historicised.

Evolver uses the historicised information from Profiler and, thanks to Data Adapters and Evolution Driver Fusion, tries to extend the database on which to build the adaptation process. Evolver generates optimised adaptation logic and updated profiling models via the specialised software modules (Fitness Evaluator and Interpolator, UI and BL Evolution Module and Model Updater), using an interactive process comprising the following steps:

*evaluation of effectiveness of previous solution – updating of models and logics – user feedback acquisition evaluation of effectiveness of previous solution - ...*

Lastly, Deployer avails itself of a Service Provider to compose the service on the basis of instructions provided by Evolver and session information provided at run-time by Profiler. Basing itself on a level separation paradigm to compose the service (content, business logic and presentation), Evolver uses traditional and "intelligent" content, retrieved respectively from a Conventional Content Provider and an Entity Provider and suitably adapted by a Content Filterer and Adapter. While, the generation of user interfaces, is referred to the UI Generator functional model which operates on the basis of optimum presentation patterns generated by Evolver. Lastly, a specific business integration block is provided for with regard to interfacing with the corporate back-end.

In short, the SAPI architecture is able to compose services, adopting a content level separation logic, presentation and business logic which allow for optimisation of the structural components of user-system interaction. This obviously represents one of the more innovative aspects of SAPI which aims, in an automatic manner, to look for the optimum adaptation solution for user and context requirements.