



AMBIENT ASSISTED LIVING, AAL

JOINT PROGRAMME

ICT-BASED SOLUTIONS FOR ADVANCEMENT OF OLDER PERSONS'
INDEPENDENCE AND PARTICIPATION IN THE "SELF-SERVE SOCIETY"

D3.1

System Conception and Architecture Design

Project acronym: **GeTVivid**
Project full title: **GeTVivid - Let's do things together**
Contract no.: **AAL-2012-5-200**
Author: **PLUS, EVISION, IRT, ISOIN, USG**
Dissemination: **Public**

TABLE OF CONTENTS

1.	EXECUTIVE SUMMARY	6
1.1	INTRODUCTION AND COMPONENTS OVERVIEW	6
1.2	AGILE SOFTWARE DEVELOPMENT	7
1.3	LINK WITH THE OBJECTIVES OF THE PROJECT	7
2.	SYSTEM ARCHITECTURE	9
2.1	ARCHITECTURE DEVELOPMENT	9
2.2	FIRST DRAFT	11
2.2.1	<i>Iteration of the initial architecture sketch</i>	<i>11</i>
2.2.2	<i>System-wide authentication</i>	<i>12</i>
2.3	SECOND DRAFT	13
2.4	THIRD DRAFT	15
2.4.1	<i>Client-server communication</i>	<i>15</i>
2.4.2	<i>Authentication and authorization.....</i>	<i>16</i>
2.5	ELABORATION AND IMPLEMENTATION OF THE THIRD AND FINAL DRAFT	19
2.5.1	<i>REST APIs</i>	<i>19</i>
2.5.2	<i>Authentication and authorization.....</i>	<i>20</i>
2.5.3	<i>Used software.....</i>	<i>22</i>
3.	SYSTEM COMPONENTS	23
3.1	APPOINTMENT COORDINATION SYSTEM (USG)	23
3.1.1	<i>System conception</i>	<i>24</i>
3.2	PROFILING SYSTEM (ISOIN)	25
3.2.1	<i>Profiler system description and integration.....</i>	<i>26</i>
3.2.2	<i>Profiling internal architecture.....</i>	<i>26</i>
3.2.3	<i>Data mining algorithms.....</i>	<i>27</i>
3.3	ADDITIONAL FUNCTIONALITY (PLUS)	28
3.3.1	<i>Conception</i>	<i>28</i>
3.3.2	<i>REST API.....</i>	<i>29</i>
3.3.3	<i>Message service.....</i>	<i>34</i>
3.4	HBBTV CLIENT (IRT).....	36
3.4.1	<i>Platform high level architecture</i>	<i>36</i>
3.4.2	<i>Technical aspects.....</i>	<i>36</i>
3.4.3	<i>Second Screen Framework.....</i>	<i>38</i>
3.5	MOBILE CLIENT (EVISION).....	39
3.5.1	<i>Client conception</i>	<i>40</i>



3.5.2 Architecture overview of the mobile client 40

3.5.3 System overview without synchronization..... 41

3.5.4 System overview with synchronization 42

4. OVERALL CONCLUSION 44

REFERENCES 45

ANNEX A 46

ANNEX B - USER STORIES 51

TERMINOLOGY & ABBREVIATIONS

- ACS Appointment Coordination System.
- AJAX call Asynchronous JavaScript + XML is an asynchronous HTTP(S) call, implies no reload of the browser window content (no blank screens or error codes). The transferred data is encoded as JSON or XML.
- API Application Programming Interface. It specifies how applications interact with each other.
- BOSH..... Bidirectional-streams Over Synchronous HTTP is an interface that allows XMPP communication via HTTP and HTTPS.
- Cake PHP CakePHP is a free, open-source, rapid development framework for PHP. It's a foundational structure for programmers to create web applications.
- CalDAV Defines that the iCal format are used to structure calendar information and how it may be transferred to, queried from, modified on, or deleted from a CalDAV server.
- Cross-Origin..... If the JavaScript tries request a url outside of the (sub)domain it is located on, then it's called cross-origin or cross-domain call. It is usually forbidden by the browser. There are some exceptions, though, like calls to JSON resources.
- CSS..... Cascading Style Sheets is a style sheet description language to define the style of HTML elements (<tags>).
- E.g..... Example given.
- Git / SVN..... Linus Torvalds distributed revision control system, named git (after himself;), which is British English slang meaning "unpleasant person". The man page describes Git as "the stupid content tracker"/Apache

SubVersion are programs that help to store different versions of files on servers and clients.

HbbTV.....	Hybrid broadcast broadband TeleVision.
HTML.....	HyperText Markup Language is a markup language to structure information.
HTTP.....	HyperText Transfer Protocol is a protocol used by browsers and web applications to get or post information in the world-wide-web.
HTTPS.....	Like HTTP, but the data is sent encrypted.
JavaScript.....	A programming language that runs on HbbTV-ready devices, HTML5 mobile applications and every modern web browser.
Jersey JAX-RS.....	Java API for XML-based RESTful Web Services is a framework that allows fast implementation of a REST API.
JQuery.....	Javascript Query library is a framework that allows to write browser-independent JavaScripts.
JSON.....	JavaScript Object Notation. Open standard that uses human readable text to describe objects for data exchange (usually) between client and server.
OAuth 2.0.....	Open Standard for Authorization 2.0. One server stores credentials, other servers or clients (services) may ask it, if the user is authenticated and authorized to use a certain service. The big idea: A user only need one account to use multiple services (e.g. log-in to Spotify with your Facebook account).
PrK.....	Private Key.
Prosody.....	XMPP server implementation. Used in this project.
PuK.....	Public Key.
RAML.....	RESTful API Modeling Language is a YAML-based language that describes RESTful APIs.
REST.....	Representational State Transfer, basically a mapping from a HTTP address (URL) to an object (e.g. calendar entry) or a process (like authentication or creation of something on the server-side).
SQL.....	Structured Query Language, a language that specifies how to query data from a SQL server (e.g. MySQL).
SSF.....	Second Screen Framework
STB.....	Set-Top Box.



- Strophe.js A JavaScript library that implements the XMPP protocol.
- Test-driven Development One first writes the tests and then starts to code. One is finished if code passes all defined tests.
- Unit Tests Every piece of code is tested by an automated test, implies that every little change in the code base has to pass the unit testing.
- User-driven Development User first explains her expectations and then one starts to code. Development is finished if code passes all defined user expectations.
- Wire Frame A crude design sketch.
- XML Extensible Markup Language is a format that is machine and human readable. Like JSON, it is used to exchange data between client and server.
- XMPP Extensible Messaging and Presence Protocol defines client2server and server2server communication, message formats etc. for instant and multi-user messaging.

1. EXECUTIVE SUMMARY

This deliverable gives an overview in terms of system architecture and its subcomponents. Therefore, only information necessary to understand the decision making process is presented. More in-depth technical details and further documentation of the components can be found in the deliverable dedicated to each system part (D3.2, D3.3, and D4.1).

1.1 Introduction and components overview

GeTVivid is a challenging project for software and system developers and software/system architects. For example, the user-driven approach on one side demands flexibility from the development team and the developed software modules, and as GeTVivid targets a specific end-user group (older adults with or without mild impairments) that demands an easy-to-use, yet stable and secure system. Challenging is also to provide an intuitive cross-platform User Management, handle the complexity of user-handling of the HbbTV Second Screen Framework (SSF) (e.g., to “log-on” should be as simple as possible and as secure possible to the system).

In order to meet this requirements intense discussions happened between the team partners about which components can be integrated in which kind to both remain flexible, when user demands change, and as secure as possible, because, of course, the system/software components need to communicate over insecure Internet communications channels sensible, if not sensitive information upon and about most user interactions. The results of iterative system design and resulting architectural choices is detailed later on.

The competence and (existing or to be expanded/developed) components of the team partners formed the basic starting point for the discussions and are sketched here in the following very briefly and also simplified to provide basic understanding of GeTVivid and its componential structure:

- User and Task Management is central for a distributed communication system as for GeTVivid. All user authentication, registration, authorization, and database - in general management - functionality is provided by the ACS component, which is maintained and further developed by USG. Also the task management interface is provided by the ACS component.
- HbbTV and SFF are the core features of GeTVivid when it comes to user interaction, where older adults prefer TV sets and mid-sized tablets. The SFF component is maintained and further developed by IRT. The TV interface is also developed and provided by IRT.
- Mobile interfaces to provide the SFF interface on tablets (for the typical older adults) and smartphones (e.g., for young(er) relatives that want to stay in touch with their grandparents) are developed and provided by EVISION.
- One key, yet practically invisible component will be responsible for the easy-of-use of GeTVivid service, when it comes to user input. Whenever a user has to input something or has to take choices, each and any suggestion (even proposing predictive assumption) are calculated and provided through the profiling system component maintained and provided by ISOIN.
- Finally all of the above components need to communicate and exchange messages, coordinate and synchronize, and have therefore the need for reliable and secure connections, which is the task of the core component provided and developed by PLUS.

In the following it is detailed how the user-driven development approach has finally been implemented in terms of development paradigm and system/software update cycles in order to provide have as much flexibility as possible during the development phase.

1.2 Agile software development

All partners agreed that an agile development schema like Scrum will be used for the GeTVivid platform. This agile approach is also reflected in the architecture, which has to be flexible enough to deal with all sorts of changes that have to be possible and easy to apply.

The development is structured in so called Sprints. The requirements are defined in the form of User Stories (see Annex B). Each User Story describes a functional feature of the platform from an end-user's perspective. This means that the detailed technical requirements and interfaces have to be defined elsewhere. For that purpose each technical telco starts with a Sprint Review and ends with a Sprint Planning session. During each technical telco tasks for all technical partners are defined. Those tasks are to be completed until the next technical telco. The time span between two technical telcos, therefore, defines the length of each Sprint (typically one month). After a User Story was selected (based on priority) all technical partners discuss necessary APIs and schedule additional telcos to work out the details. The Sprint Reviews are used to reflect the development progress.

1.3 Link with the objectives of the project

A good fundament is the base for every successful building. That said, the development of a suitable solution for the architecture design is the fundament on which all the technical developments are based upon. Before the actual architecture was decided upon, there were several rounds of brainstorming and various ways of approaching the actual tasks. More about these steps can be found in sections 2.1. to 2.4. See an example of a brainstorming table in Figure 1.

The user and business requirements from T2.2 and T5.2 are transferred into technical requirements for the soft- and hardware development in the WP3 and WP4. This is done by defining high-level scenarios and use-cases (see Annex B / User Stories), which is the usual approach in a Scrum-like development framework. This forms the basis for the definition of the system architecture and the necessary components. Furthermore, a detailed system design is developed in order to specify the communication flows, first interfaces and protocols. The end-user device (TV, STB, mobile, tablet, etc.) specifications are done in this task. They can be found in sections 3.4 and 3.5 or in detail in the corresponding deliverable (D3.2 and D4.1).

The latest prototypes are hosted on <https://demo.getvivid.eu/> and <http://getvivid.amiona.eu/>.

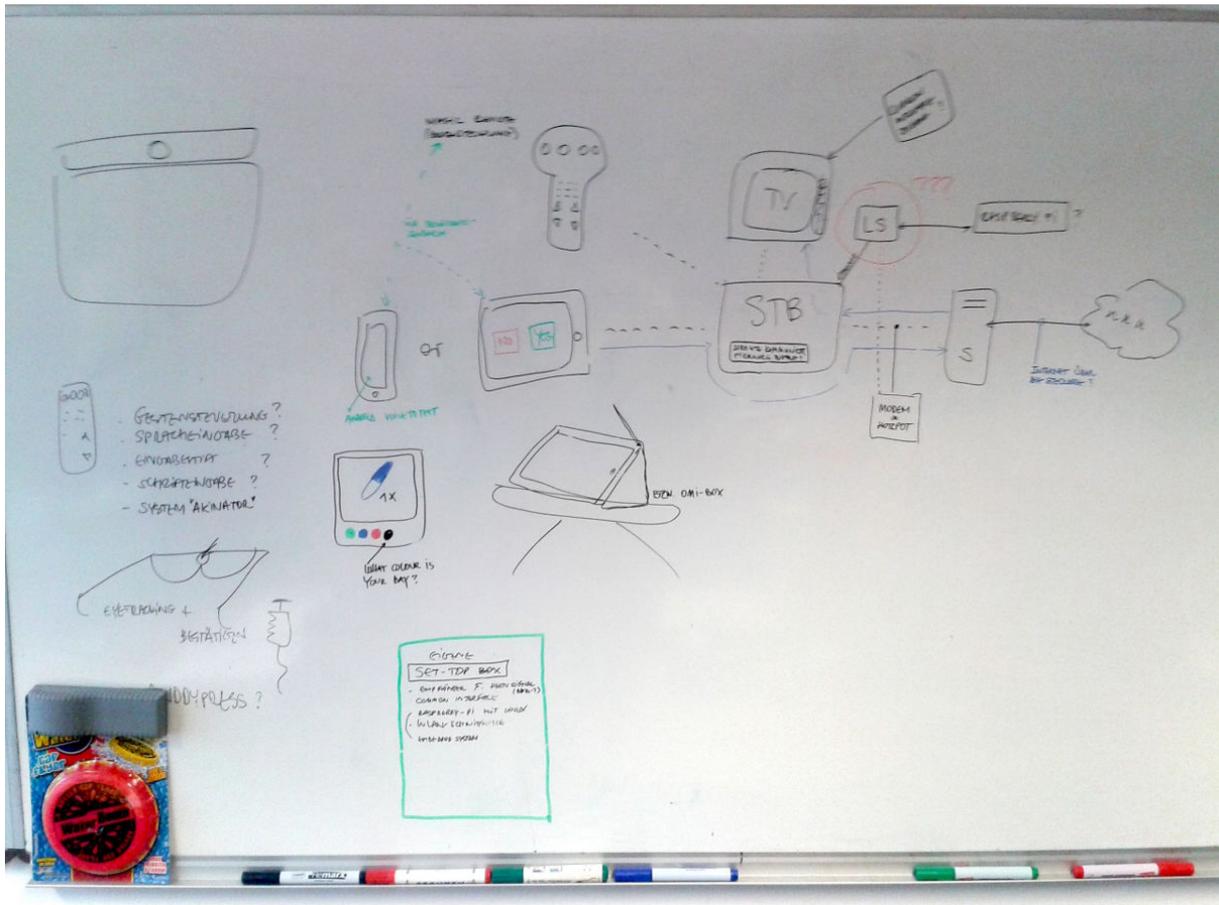


Figure 1: Brainstorming sketches of possible architecture and devices

2. SYSTEM ARCHITECTURE

This section documents the development process of the GeTVivid platform's system architecture. Following subsections describe the iterations, which were necessary to find a suitable architecture.

SCRUM has one big potential disadvantage for the system architecture, which is that in the end parts of the architecture might be more complicated than they have to be, but we will always keep an eye on this (i.e., changes introduced by the feedback of end-user evaluations or by business requirement analysis will be carefully reviewed).

2.1 Architecture development

Even before the actual start of the project, the soon to be members of the consortium were discussing the possible architecture of the concept. Few but important points were identified, defined and found their condensation in the first draft of the desired system architecture (see Figure 2). There will be multiple parties involved. The end users can be categorized into the older adults, the relatives and professional service providers. Business partners of the platform itself will most likely be suppliers of external information channels and broadcast companies, which have to deliver the HbbTV content to the end-user's home and TV set.

Starting from the first glimpse the basic team / consortium developed further ideas and concrete conceptions of how the basic architecture could look like – still not taking into consideration the state of the art products and technology available today. A crude separation into system components and dedicated frontends for different end-users are already visible in Figure 3. A TV and a Mobile client constitute the frontend for the main system users, the older adult. Separate frontends are already defined for professional service providers, relatives and the platform operator. There were already several decisions that can be made based upon this plain sketch. Firstly, both TV and Mobile client will have to perform very similar tasks. Both could use the same API to communicate with the backend.

Secondly, there are several configurations for the end-user's interaction with the system. Those configurations have later been defined as profiles. D3.2 describes those profiles in detail. It defines the Broadcast Profile (no Internet connection; one-way), the Interactive Profile (Internet connection and broadcast) and the Interactive Profile with a Second-Screen (e.g., a tablet). It was decided that the primary development focus will lie on the latter, because it offers the highest amount of functionality and is therefore the most complicated to implement. The previous two profiles are more or less functional subsets and can be derived from the Interactive Profile with Second-Screen.

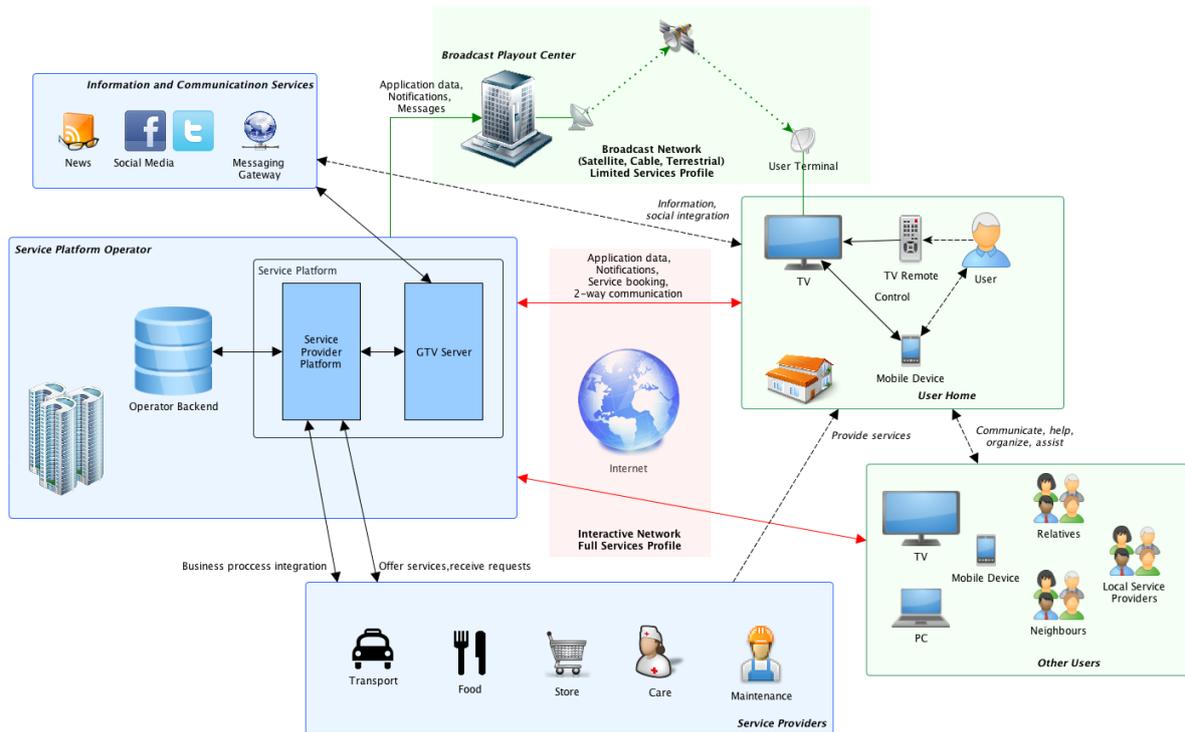


Figure 2: The first idea of a future concept and architecture

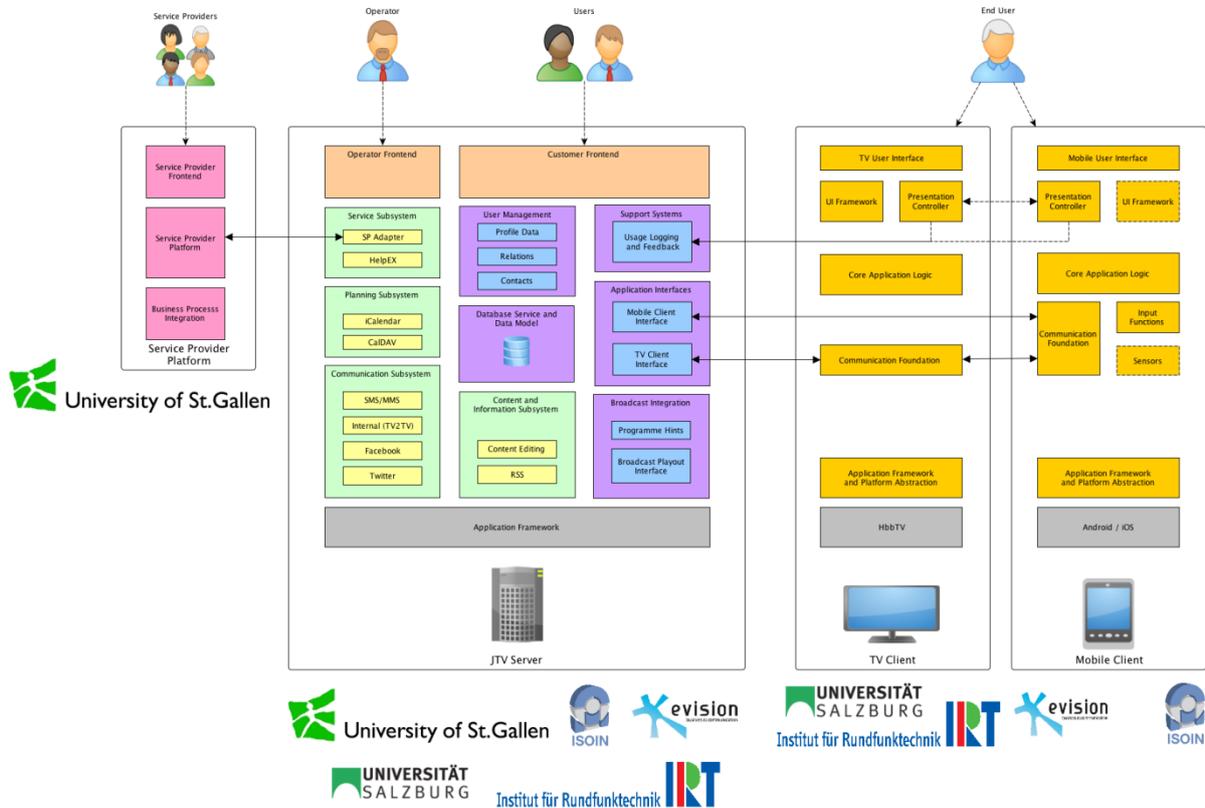


Figure 3: Logic layout and aggregation

2.2 First draft

The first draft is the initial attempt to combine all technologies used by the project partners in their existing components. Those components include: the Appointment Coordination System (ACS, developed by USG), the Second-Screen Framework (SSF, developed by IRT and used for Second-Screen Interaction) and existing open-source software, which would fit the project's needs. A top-down analysis of the involved components led to some rather obvious technological choices:

- Since the SSF runs as a JavaScript Library it makes sense to implement the mobile application as HTML4.1/HTML5 application. Furthermore, the HbbTV standard only supports a subset of the HTML Standard (CE-HTML) and again JavaScript, so the communication via HTTPS with the server is the feasible and safest solution.
- This implies a HTTP interface on the server side. At the Kick-Off Meeting the use of JSON as exchange data format was discussed, which seemed to be best alternative, since JavaScript and most other programming languages offer native functions or methods to encode and decode JSON.
- Figure 3 shows that all backend components unrelated to appointment coordination can be developed separately from the ACS. Still, it was unclear if the other sub systems could be developed independently. The lack of detailed requirements slowed down the decision process.
- PHP is proposed by PLUS for initial rapid prototyping of possible backend features. JSP and Ruby On Rails were also discussed shortly but were disqualified. ISOIN suggested CakePHP as framework, because they had positive experience in past projects.

2.2.1 Iteration of the initial architecture sketch

The product of the initial decisions, the first logical architecture sketch is displayed in Figure 4. For clarification, the colours only signalize logical groups. Dark red stands for external services, which may or may not be used by the GeTVivid application. At this point it is not predictable, if there will be a need to access the APIs of social networks like Facebook or Google+. Grey are software components and implementations that are already available. These will be discussed below. Green symbolizes the GeTVivid Core Logic (roughly based on Figure 3).

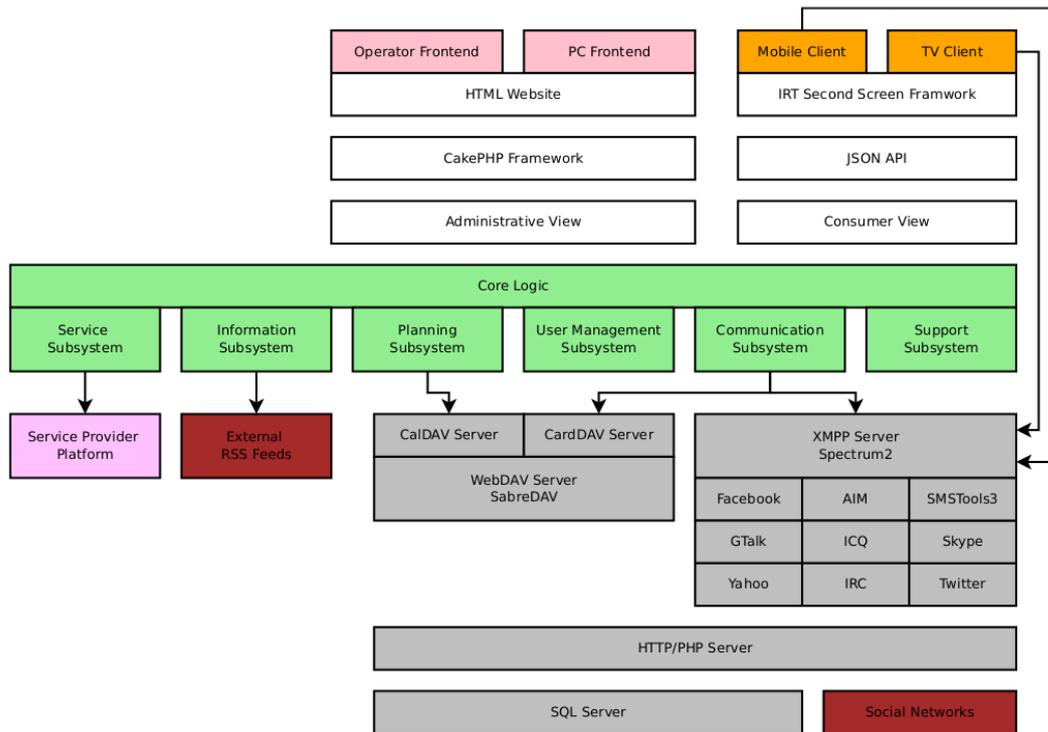


Figure 4: Architecture draft - a functional hierarchy of possible system parts.

For rapid prototyping it would make sense to implement the whole Core Logic as CakePHP application. It is easily to extend with functionality and already contains basic primitives for permission and user management. CakePHP uses the Model-View-Controller (MVC) design pattern, which would allow to use the same models for the JSON API (used by TV and Mobile Client) and for the Operator/PC frontend. The different interfaces would be implemented as different Views. This design concept would also enable the developers to bring the administrative interface to mobile devices by simply replacing the Views. This was the state of the architecture in September 2013.

2.2.2 System-wide authentication

At the technical meeting in Munich in January 2014 the integration with the ACS was discussed. It was soon clear, that the ACS user database was the point to start and that the first step would be to find a feasible solution to authenticate users in the system in a secure manner. The biggest challenge will be the Cross-Device-SSO, which is necessary for a good user experience. Figure 5 displays the first idea of how to implement this feature. It is not elaborated here, because after another discussion OAuth 2.0 was considered a possible standard to use for user authentication. It was decided that the ACS will provide an OAuth 2.0 authorization server for the rest of the system.

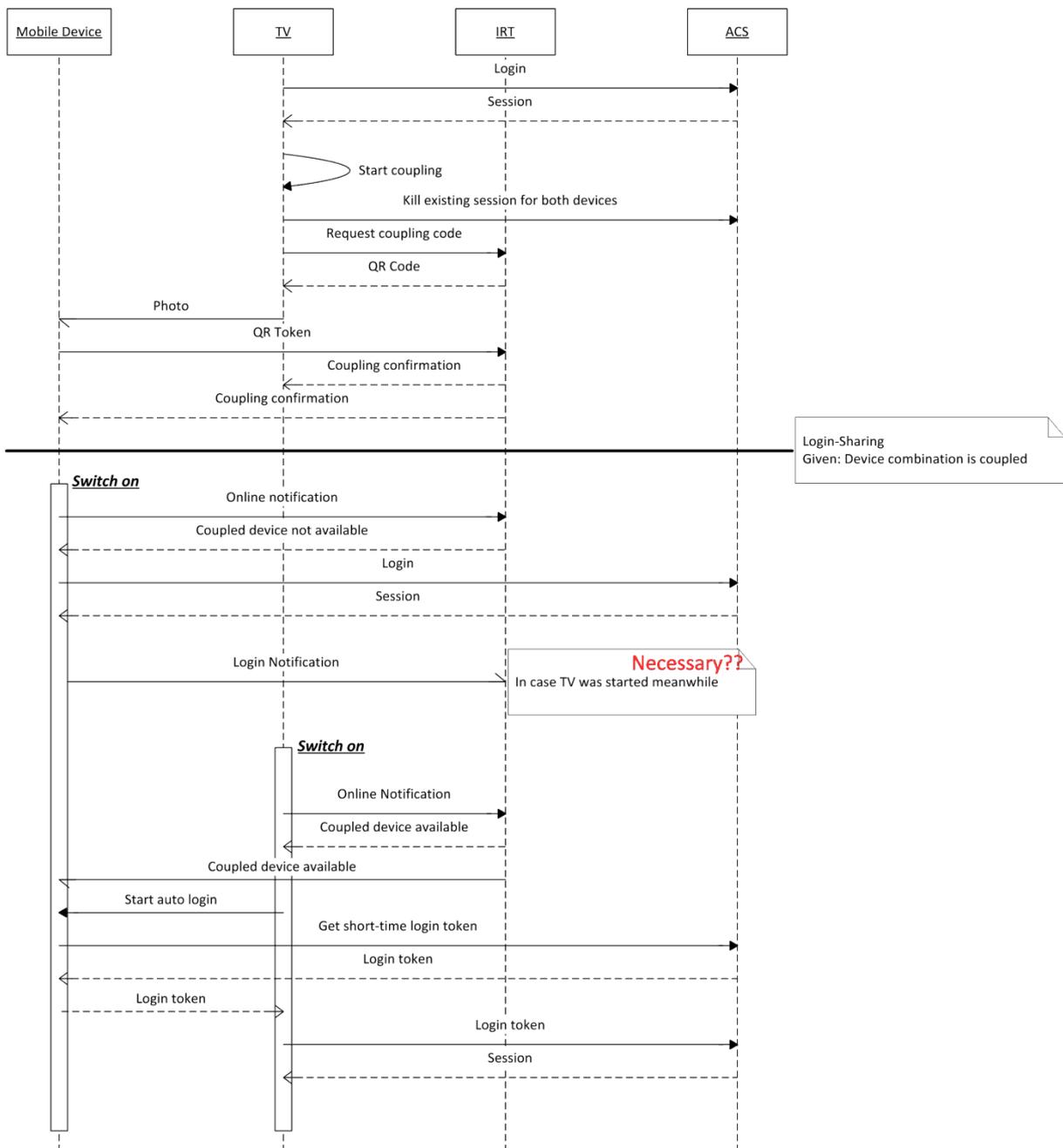


Figure 5: Second-screen login

2.3 Second draft

After some research on available software libraries and components PLUS came up with following proposal for the technical project meeting in March 2014. Figure 6 shows a possible logic representation of the proposed changes to the first draft. This iteration already integrates the OAuth2.0 authorization logic and communication paths. Furthermore, a REST API is suggested as interface for all client applications (Mobile, HbbTV and Web client).

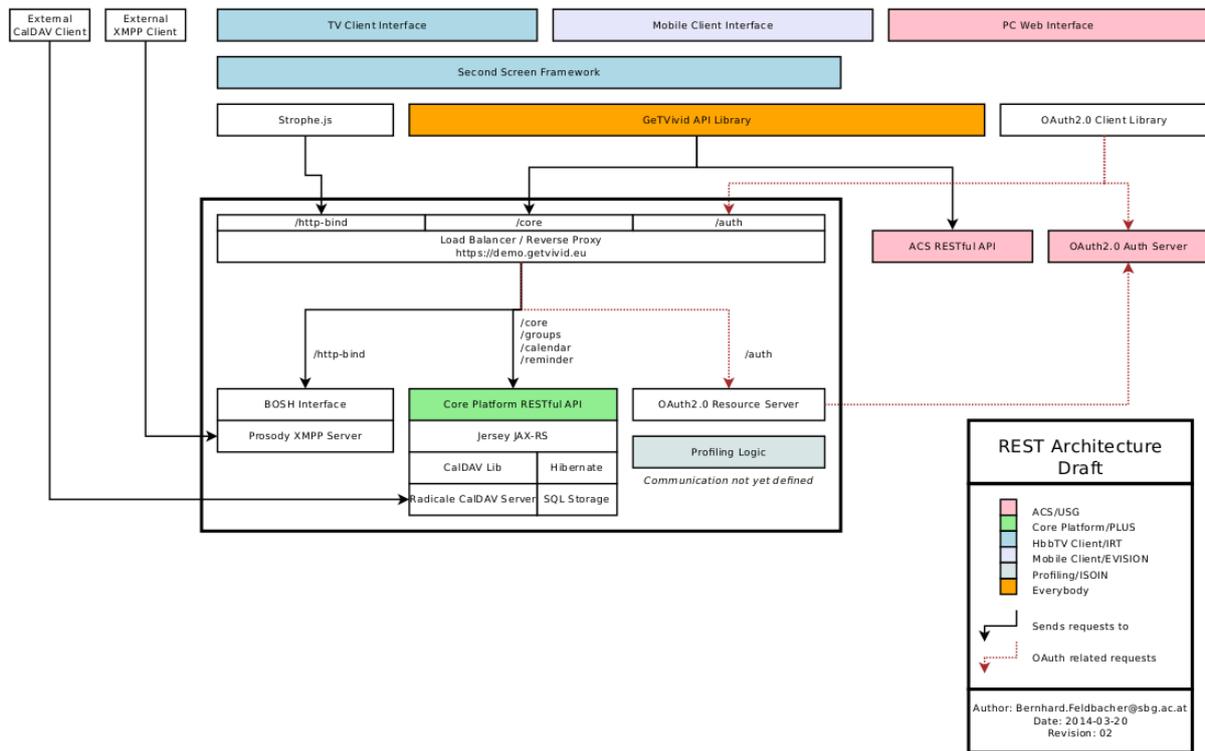


Figure 6: Proposed components

Here is a list describing the most **important characteristics** of the proposed architecture scheme:

- HbbTV and Mobile client are responsive web/HTML applications. After the applications are downloaded to the client all communication takes place via asynchronous HTTPS calls. The benefits are a user friendly error handling (no empty screens or error codes) and a clear separation of server-side business logic and client-side user interface code.
- For the server-side PLUS proposes a REST API, which can be used by the HbbTV, Mobile and even Web client.
- The Web client is also a responsive web application. It is jointly decided to use the existing user interface of the ACS and modify/extend it with the required additional functionality. Communication with the core platform logic is therefore accomplished with AJAX calls, which again makes use of the REST API.
- The communication between ACS and core platform are implemented as calls to each other's REST API.
- Since the ACS already has a user database/management it is obliging to store the central user database on the ACS server. Authentication and authorization are therefore also managed by the ACS. OAuth 2.0 meets the basic requirements for our client-server architecture.
- For instant messaging, mailing-list functionality, multi-user chats etc. XMPP are used. The existing libraries are well tested and allow an easy integration into the client-side code. The communication with HbbTV, Mobile and Web client will make use of a BOSH interface (HTTPS communication).

- For the storage of calendar data (events, todos, reminders etc.) CalDAV could be an option. It supports complex calendar information and access through external clients or synchronization with external sources. Various implementations for the CalDAV standard already exist. Storage, modification, filtering etc. are therefore already implemented by the existing libraries.

Benefits of this architecture are:

- The server-side business logic of core platform or ACS may be changed without interfering with the REST interface and client-side code.
- A REST API allows the integration of third-party services (apps inside the system). XMPP and CalDAV allow communication with external clients and resources.
- A REST API may be tested separately from clients and user interface.
- The consequent usage of the same techniques (REST API, OAuth, XMPP etc.) allows code reuse on the client-side. There will be only one library for OAuth2.0 and one to access the core platform logic/ACS logic on the client-side.
- Changes in the user interface design will not affect the server-side code and may be executed by the user interface developers on a separate code base.
- All three clients will make use of nearly the same technologies (HTML, CSS, JavaScript), hence know-how transfer is easy and possible.

The draft and most of the proposed technologies were discussed during the technical meeting in March 2014. The CalDav components were replaced with the calendar of the ACS since it already used the iCal standard. The XMPP standard was new to most of the technical partners. Thus, it needed further investigation, before a final decision was made (see next section). The rest of the design was considered a step in the right direction.

2.4 Third draft

2.4.1 Client-server communication

Based on the discussion in the meeting in March another draft was created which takes care of the cross-origin problem that HTML applications usually have (see Figure 7). The idea of a HTTPS broker or proxy was introduced in order to create one backend for client-server communication. Furthermore, a reverse proxy kept the architecture quite flexible, which was still necessary, because most of the interfaces were undefined or the subject of current discussions.

A dedicated development and demo server was hosted by PLUS equipped with the necessary certificates to create a secure communication channel with the clients. The green arrows in Figure 7 show requests made by the Mobile client. Same goes for the HbbTV client in blue and the Web client in yellow. All connections from and to demo.getvivid.eu, the demo server, are secured with SSL encryption. One security question remaining at this point is the SSF communication channel. The SSF is hosted by the IRT and is only available as a service. It therefore cannot be further integrated into the proxy architecture. The SSF uses JSONP to avoid cross-origin

errors. As this way of communicating is not as common as one might think, it is not entirely clear how secure it is. For that reason JSONP will be further investigated regarding security issues. Also other possibilities like point-to-point encryption will be discussed. In any case, the channel will be encrypted in one or the other way.

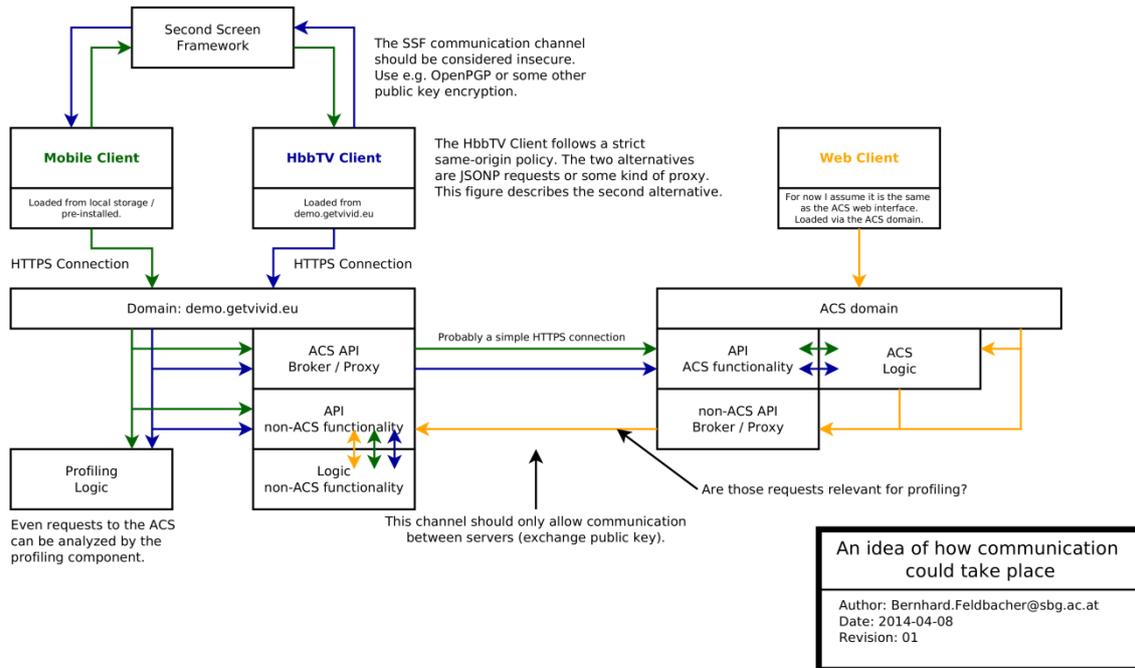


Figure 7: An idea of how client-server communication could take place

2.4.2 Authentication and authorization

Figure 8 displays an authentication and authorization scheme based on OAuth 2.0 proposed by PLUS, which is elaborated below. Basically the client (HbbTV or Mobile client) performs a log-in on the OAuth2.0 authorization server and receives an Access Token at the end of the procedure. This Access Token is then used by the client to access (REST) APIs provided by the resource servers (see components in section 3 of this document). The figure below tries to visualize which components have to communicate to allow only authorized users access to protected resources.

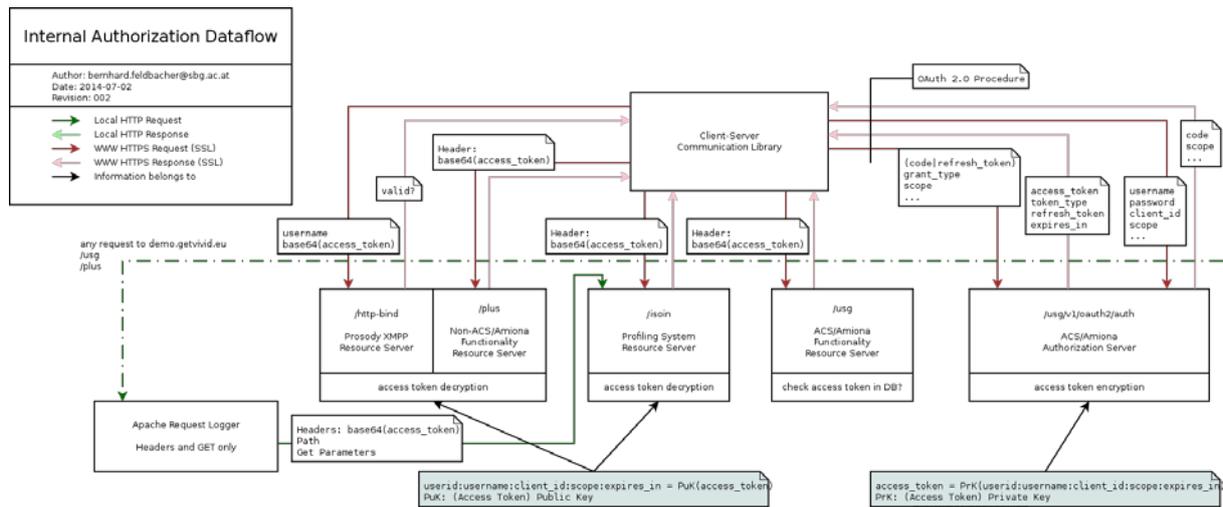


Figure 8: OAuth 2.0 components with information encoded in access tokens.

The components shown in Figure 8 were analysed with following outcome:

Authorization server: One central OAuth 2.0 authorization server takes care of the user authentication and authorization. For the initial phase of the project the authorization server only knows one authorization scope: DEFAULT. This server is provided by the ACS platform.

Clients: In the current system draft two OAuth 2.0 clients, one for mobiles and one for HbbTV conform devices, communicate with the GetVivid platform's servers.

Permanent log-in: To allow a permanent log-in, at least one of the clients would have to store the refresh token (which is valid until the next access token is requested). An alternative would be to store username and passwords in the client's memory, but this is clearly not an option, because this would expose the user's credentials to certain attacks.

Log-in sharing

- Option A: Log-in sharing can be accomplished by sharing the temporary access token. Since both clients should be as independent as possible, the refresh token has to be shared too. The refresh token is special, since it may only be used once. Thus, this implies that one of both clients has to act as master, which retrieves a new access token once the old one has expired.

The token exchange could look like this: The first device turned on uses the stored refresh token to retrieve an access token from the authorization server. The second device on-line has to ask the first one via the SSF if a valid access token is available. The first one responds with the access token AND the refresh token. Once one of both clients is turned off, the other one is able to take over the position of the master client and take care of the token refresh business.

Following issues are not resolved by this option:

- What happens if the retrieval of a new access token fails?
- What happens if the master is turned off before it can communicate the new refresh and access token?

- What happens if the master is off-line while the former slave is turned on? It would still have an invalid refresh token in its cache.
- Option B: Implement a server-side API to copy the session of the current user to another client. An easy solution is to simply respond to a token refresh request with two refresh tokens if a certain request parameter is given. The second refresh token is sent via the SSF to the second client. Security measure: Only allow two valid refresh tokens per user at each point in time. Once the refresh token was successfully duplicated most of the issues mentioned in option A are resolved. Even if the retrieval of a new access token fails, the client can still request a new refresh token from the second client. Furthermore, option B seems to be the development-wise most efficient solution.

Resource servers

There are several distributed OAuth 2.0 resource server involved in the GetVivid platform. Those are ISOIN's profiling sub system, USG's ACS service provider platform and PLUS' resource server application for all additional functionality. Apart from the authorization system there is also IRT's SSF server, which does not need any valid ACS session. Therefore, it is not considered as an OAuth 2.0 resource server. Each resource server needs to know

- if a given access token is (still) valid (expired),
- to which user a given access token was issued (to store user specific data),
- and which scopes are accessible with a given access token?

Again, there are two possible approaches:

- Option A: With the current implementation an API to get the information mentioned above has to be specified (from the authorization server). Of course, such a request slows down the whole system, if every request to a resource server results in another request to the authorization server. Thus, it is necessary to store each access token in each resource server's database. This introduces following problems:
 - The risk of potential data leaks is increased in case any of the resource servers is compromised. Each access token allows access to all parts of the system (at least to the parts specified by the scope parameter and for the specified time frame). From the security perspective an access token is therefore nearly as sensitive as the user's credentials. Thus, the access tokens should be only stores as hash values.
 Following issues are not resolved by option A:
 - Each new access token causes at least one additional request to the authorization server issued by each resource server accessed with this token.
- Option B: Encode the corresponding user id and user name, client id, scope and expiration date-time into the token. This can be accomplished by using a public key encryption algorithm. All resource servers may share one public key for decryption. The authorization server uses the private key for encryption. The Base64 algorithm can be used to transform the resulting binary data into an ASCII string and vice versa.

Implications:

- There is no need for token validation, which happens implicit if the token can be decrypted with the public key.
- There is no need for the resource server to store the tokens locally. This eliminates the risk of a potential data leak.
- Protected resources are as secure as the private key.

Although option B is the solution with less traffic overhead, it is the more expensive implementation regarding development costs. See the next section for a final decision.

2.5 Elaboration and implementation of the third and final draft

2.5.1 REST APIs

The final draft of the REST communication architecture introduces a client-server communication library, which provides a JavaScript interface that can be used by both the Mobile and the HbbTV client.

The final service setup is described below:

- **The reverse proxy:** Due to the nature of REST APIs (as implemented by the ACS API and as proposed by the Profiling API) different HTTP methods are needed for communication (POST, GET etc.). In previous telephone conferences JSONP was discussed for client-server communication, which would only allow HTTP GET requests. Thus, a reverse proxy is the next best alternative. Furthermore, the reverse proxy can be updated with a load balancer, if necessary and it can also take care of SSL encryption of the client-server communication for all services hosted on the same host.

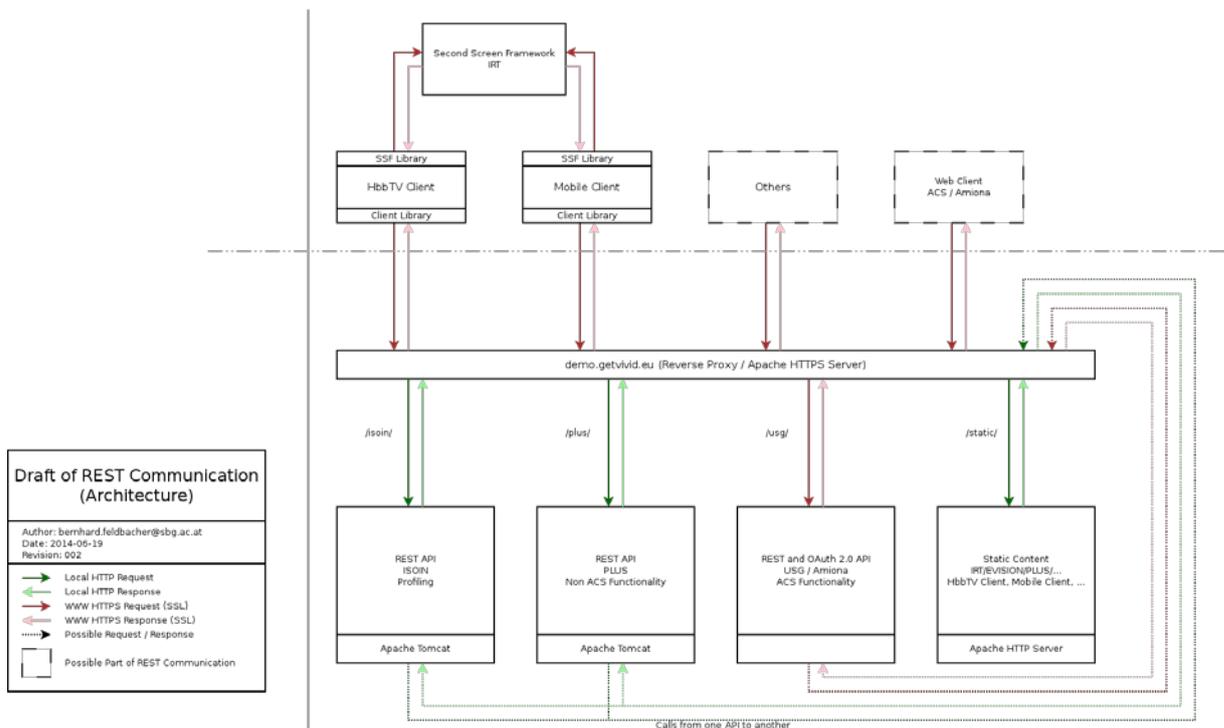


Figure 9: Final version of REST architecture

PLUS installed an Apache Reverse Proxy, which maps the incoming requests based on path prefixes (/isoin, /plus, /usg, /static) to the different back-ends. A call to the ACS API looks like: <https://demo.getvivid.eu/usg/v1/oauth2/auth>. Right now it is not decided if the Web Client will use the REST APIs too. But it should be pointed out, that this architecture allows extending the GetVivid platform with other client applications without much overhead.

Forwards marked with green colour will be local forwards (see Figure 9). Red colour indicates a remote connection which will be secured by HTTPS. In the technical tel. Conference in June 2014 all attending partners agreed to this setup.

- **Profiler REST API** (see section 3.2 for details): According to the documents presented in March 2014 the Profiling API is protected by HTTP Basic Authentication and will only be accessible to other backend components. The communication will take place via HTTPS. The profiler runs on a dedicated Tomcat instance on demo.getvivid.eu.
- **ACS REST and OAuth 2.0 API** (see section 3.1 for details): The ACS provides an OAuth 2.0 API. Endpoints necessary for the OAuth 2.0 procedure are again forwarded by the proxy to avoid any cross-origin errors. For further details see the section 3.1.2.5. A dedicated GetVivid ACS instance runs on a server provided by USG
- **Additional functionality REST API** (see section 3.3 for details): PLUS develops all endpoints necessary for integration and for functionality not covered by the ACS. Communication will again take place via HTTPS. It will run on a dedicated Tomcat instance on demo.getvivid.eu.
- **Second-Screen framework** (see section 3.4.3 for details): The SSF is hosted as a service by the IRT.
- **Static content**: The host demo.getvivid.eu also provides a static HTTP server to serve static data like the HbbTV and Mobile client.

2.5.2 Authentication and authorization

The final iteration of the authorization and authentication scheme is depicted in Figure 10. All resource servers of the system use a REST endpoint to verify each new access token and to retrieve relevant information to limit or allow access to certain resources. The endpoint returns following information on a given access token (see Listing 1). This version of the authentication scheme now displays detailed communication paths including the information transported.

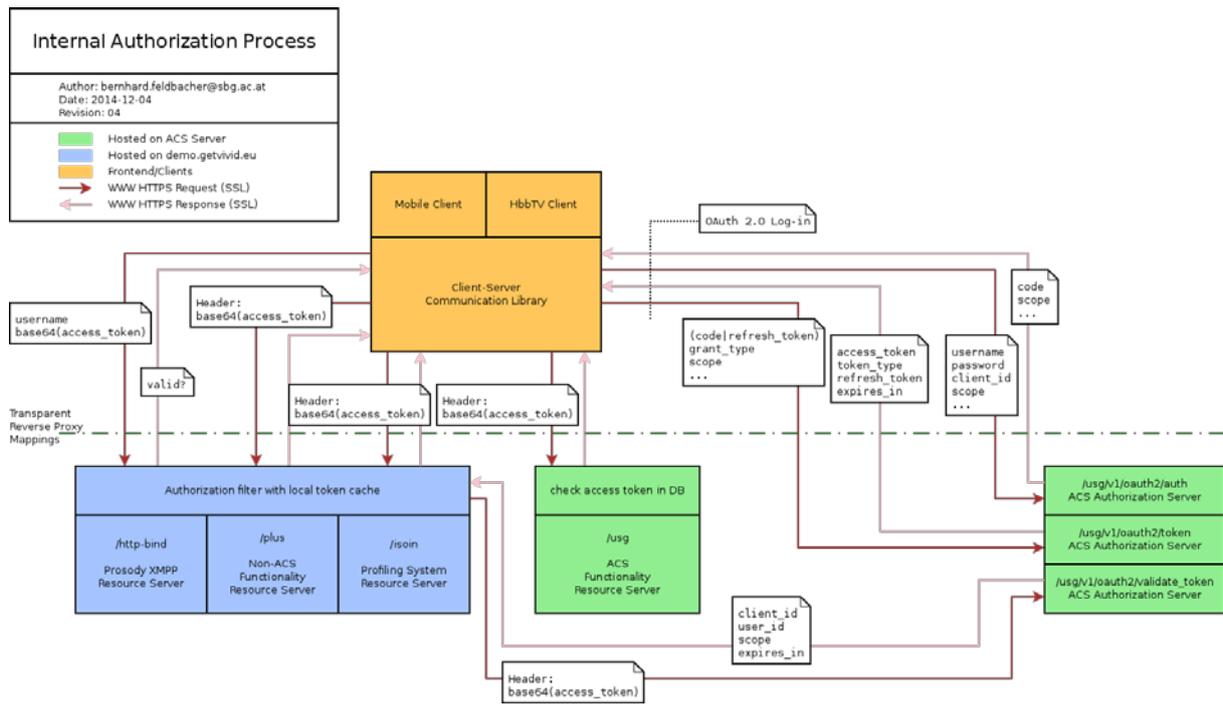


Figure 10: OAuth 2.0 components with a token validation endpoint.

```
{
  "client_id": "oapYn78sDWAbzy5NZwiy", /* for future use */
  "user_id": "123456789", /* ACS user id */
  "scope": "DEFAULT", /* for future use */
  "expires_in": 436 /* seconds */
}
```

Listing 1: Information returned by token validation endpoint

Resource server

This iteration of the scheme comes with details on the implementation: Tokens and related information are stored in the resource server's local token cache in order to minimize the number of validation requests. Once a token is expired it is automatically removed from the cache.

All requests to the profiler, the XMPP server resp. the BOSH interface and the additional functionality are handled in the form of a central implementation of an authentication resource filter (these components are filled with blue in Figure 10).

The resource server that contains all ACS features (filled with green in Figure 10) uses its own implementation of a resource protection. The web interface uses a session based log-in. The REST API has access to the OAuth 2.0 authorization server and its database.

Authorization server and clients

The ACS provides an OAuth 2.0 authorization server. HbbTV and Mobile client act as OAuth 2.0 clients. More details have been discussed in the previous section (see section 2.4.2). At this point multiple possible ways to accomplish login sharing have been discussed (see previous section). All seem feasible, but the implementation is subject of a user story, that has yet to be implemented.

2.5.3 Used software

This section lists installed software packages and configuration, which were needed to implement the architecture described in the previous sections.

- **Operating system** of demo.getvivid.eu: Debian Wheezy (7.6) with kernel 3.2.0-4-amd64 and root access.
- **Web server** (Tomcat7, Apache2): Since the HbbTV specification uses HTTP(S) for communication, a web server is needed. Apache2 is installed and configured as reverse proxy server. There is also a static web server included in the Apache installation for HbbTV and Mobile client.

At this point Java is the preferred programming language. Thus, Tomcat7 is installed and multiple hosts are configured. One instance is used for the profiling deployments and one is used for the additional functionality.

- **Certificate:** A list of root certificates valid according to the HbbTV specification can be found here: <https://www.hbbtv.org/spec/certificates.html>. Since the domain is already hosted by GoDaddy, the certificate was also purchased there.
- **XMPP Server/Prosody:** A Prosody (see <http://prosody.im/>) XMPP server with a BOSH interface is installed and configured. The ACS OAuth 2.0 interface is used to authenticate users. There is also a static contact list for all development team members, which contains each members contact. All data is stored in a **MySQL** database.

3. SYSTEM COMPONENTS

The GeTVivid platform is composed of several subcomponents, which either have been in development since before the project started or have been developed especially for the project. This section contains these components and elaborates their individual development goals. Please note that each section has been written by the partner developing the named component. Therefore, the focus and goals of the development may vary. Since two of the technical partners are companies of the private sector (EVISION, ISOIN) and the other three have an institutional background (IRT, PLUS, USG), it should be understandable that the prerequisites for each partner's contribution are very different and so is the product.

The previous section gave an idea of how the following components are connected and which means of communication are possible. The ACS (see section 3.1) is basically the core component as it includes the GeTVivid platform's central user database. The other two back-end components, the profiling system (see section 3.2) and the additional functionality (see section 3.3), rely on the ACS and therefore depend on its data structures and APIs. The two front-end components exclusively developed for the GeTVivid platform, the HbbTV (see section 3.4) and the Mobile client (see section 3.5), communicate with the backend's REST APIs.

3.1 Appointment Coordination System (USG)

Appointment Coordination System (ACS) from the Competence Centre for Independent Living at the University of St. Gallen is a hosted solution for a broad range of electronic service marketplace scenarios. Besides basic functionalities like user account management, a search function, or an iCal feed based calendar integration, the main difference between the mentioned scenarios is based on varieties in the request mode, such as:

- Consumer triggered – normal request: A request for a certain service and date (including the time)¹.
- Consumer triggered – request without date: A request for a certain service but without specification of the desired date. Providers are expected to answer such requests by suggesting 1 to n possible dates².
- Provider triggered – fixed date confirmation: An appointment is directly entered by the provider without requiring any interaction from consumers³.
- Provider triggered – multiple date suggestions: 1 to n possible dates are suggested by the provider without having a preceding request from a consumer⁴.

The following (sub-) sections give a first, brief overview about the concept and architecture design behind ACS. More details can be found in deliverable D3.3 (Service Platform).

¹ The normal form of a request which is helpful if the chance of striking a free time slot at the provider's side is not too bad.

² Useful if the provider is comparatively busy.

³ This approach can be used when a consumer is requesting a service outside of the integrated process environment or channels. The message from the provider to a consumer can be considered as a booking confirmation.

⁴ Might e.g. be used for periodically repeated services where a provider reminds its customers in order to increase customer loyalty.

3.1.1 System conception

3.1.1.1 Basic architecture

In general, there are three components respectively three layers involved when it comes to utilizing ACS (see Figure 11). The first and most important layer is the backend of the system, which encapsulates the data structure and process logic. The frontend as second layer is in comparison to the backend rather a collection of multiple user interfaces, such as the normal web-interface or, more specifically for GeTVivid, the HbbTV interface. The client layer as last tier is not necessarily part of the actual ACS or GeTVivid platform. It represents any solution that is necessary in order to visualize what has been defined on the frontend layer. For a simple web-interface this might be a browser. However, there might be additional scenarios where this layer contains, e.g., container-apps or embedded functionalities.

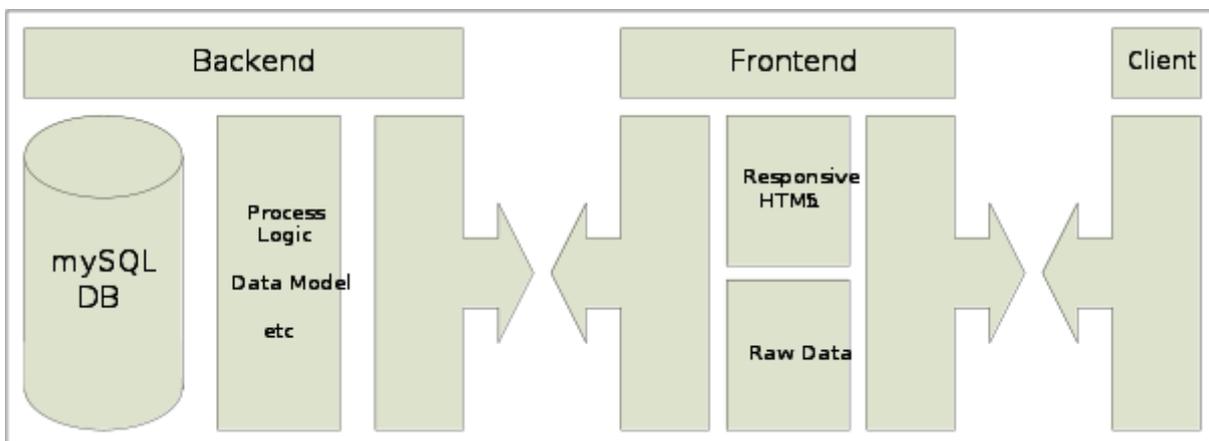


Figure 11: ACS Basic Architecture

3.1.1.2 Backend

In order to follow the principles of modularity, adaptability, and extensibility, the main components resp. layers of ACS are loosely coupled. The first and most important layer is the backend, which contains among others the main process logic of the application. Parts of this process logic might also be available in frontends in a redundant manner, but the power of ultimate decision about the validity of process steps is located in the backend. Whenever information has to be stored resp. made persisted or has to be accessed, a data object model approach is used in order to access MySQL resources.

In order to seal both data storage and process logic off, the only way to access backend resources is via REST web-services. This interface, however, is not completely open to all possible applications (and hence “frontends”) at the moment, but limited to pre-registered ones. Additional information might be found in section 3.1.2.5 as well as in deliverable D3.3.

3.1.1.3 Web-Frontend

ACS supports multiple frontends for a single tenant, whereas a tenant represents e.g. a project like GeTVivid. One frontend that is available by default is the standard ACS web-frontend. It includes workspaces for both, (professional) providers and consumers. However, in the context of GeTVivid the web-frontend will primarily be used by providers.

Generally, a frontend can be considered as an UI-building control, which may utilize all available REST web-service functionalities and translate the results resp. the data usable representations. In terms of the standard web-frontend this means, that from REST requests resulting data is transformed in HTML5 code which is provided to the consumer via webserver. Parallel to this approach, raw datasets are provided for asynchronous request that are rendered within the client application.

3.1.1.4 Client

The third layer of ACS, the client application, should be understood in a more abstract manner. In contrast to other components, this layer might in some cases just be a standard application, e.g. a web browser. However, in the context of GetVivid this might be an application (possibly even an embedded one) running on a HbbTV device and enabling channel and program independent functionalities.

3.1.1.5 ACS OAuth REST interface

The ACS OAuth interface can be used in order to implement a secure delegated access to the resources resp. the data of a certain ACS user. Since all requests have to be made on behalf of such a user, the user's credentials are required. The inversion of this argument is that the OAuth REST services cannot be used in order to perform standard platform operations like requesting the name for a given user ID (except it is the ID of the user that is represented by the OAuth access token). For more details on the system authorization process and the integration with other components refer to section 2.5.2 and D3.3.

3.2 Profiling system (ISOIN)

A profile is a description of a person with the most important or relevant information about him or her. In software applications the profile is obtained from the users to complete the user profile, also called user model containing essential information about the user. The creation of these profiles let the system know the differences between users to offer personalized services for each one.

User profiles are different depending on the domain of the application. For example, in a music website, the domain will cover the songs and groups that the user likes and also genres. In the case of a dating page, the user profile will contain other information like age, sex, hair colour, eyes colour, personality traits, and much other important attributes.

Not only the content of user profiles differs from one domain to another, but also how the information they contain is acquired. The content of a user profile can be explicitly provided by the user or it has to be learned using some intelligent technique. User profiling implies inferring unobservable information about users from observable information about them, that is, their actions or utterances. A wide variety of Artificial Intelligence techniques have been used for user profiling, such as case-based reasoning, Bayesian, association, genetic algorithms neural networks, among others.

The purpose of obtaining user profiles is also different in the various areas that use them. A few examples:

- In adaptive systems, the user profile is used to provide the adaptation effect. That is to behave differently in different situations.
- In intelligent agents, particularly in interface agents, the user profile is used to provide personalized assistance to users with respect to some software application.

- In intelligent tutoring systems, the user profile or student model is used to guide students in their learning process according to their knowledge and learning styles.
- In e-commerce applications the user or customer profile is used to make personalized offers and to suggest or recommend products the user is supposed to like.
- In knowledge management systems, the skills a user or employee has, the roles he takes within an organization and his performance in these roles are used by managers or project leaders to assign him to the job position that suits him.
- In recommender systems the user profile contains ratings for items like movies, news or books, which are used to recommend potentially interesting items to him and to other users with similar tastes or interests.

3.2.1 Profiler system description and integration

GeTVivid Profiler System provides a RESTful API that can be consumed by any REST client in order to allow integration with other modules of the GeTVivid platform. This API exposes a resource named "profiler" through GET and POST methods, receiving as input a XML document described by a XSD schema. API response includes an XML document with a useful error code\message. GeTVivid Profiler API is secured by SSL (HTTPS) using Basic Authentication. API is stateless so all request must be authenticated.

Interaction between the GeTVivid platform and the profiler module is shown in the picture below. Core functionalities of the profiler system are exposed on a REST interface and information is sent and received over XML files. The profiler system is a Java web-based application, so it must run on an application server or a servlet container as Tomcat (see Figure 12).

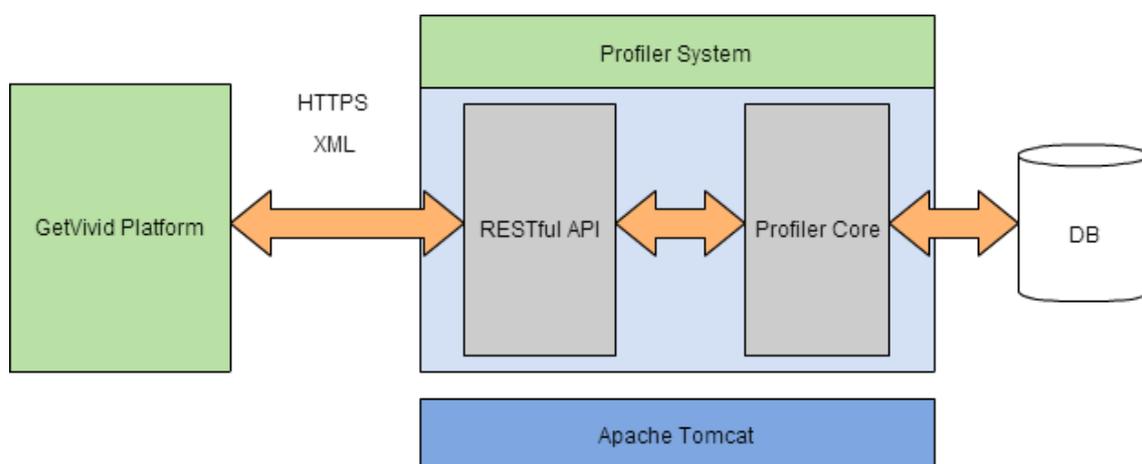


Figure 12: Profiler system

3.2.2 Profiling internal architecture

Figure 13 below shows the internal architecture of the profiler system and its interactions with the rest of the system. First you can find the API methods to put (POST) and retrieve (GET) information; these two methods are the in/out points of the profiler system. By using the POST method, the GeTVivid platform will add data to

the profiler database, which will be used to make the profiling of the user. These data are stored on the database for further and deeper analysis using data mining techniques such as K-nearest neighbour, classification and clustering.

With these techniques, profiles can be extracted from the users which let the system to create models of the users and their behaviour as well as rank the services. With all this information the system will use another module which is responsible for creating new recommendations for the users, look for similar services or look for certain user behaviour patterns.

Finally these recommendations, suggestions, alerts, etc. can be retrieved from the profiler system by using the GET method; this request must include a definition of the request service (name of the service, and required parameters) inside an XML file.

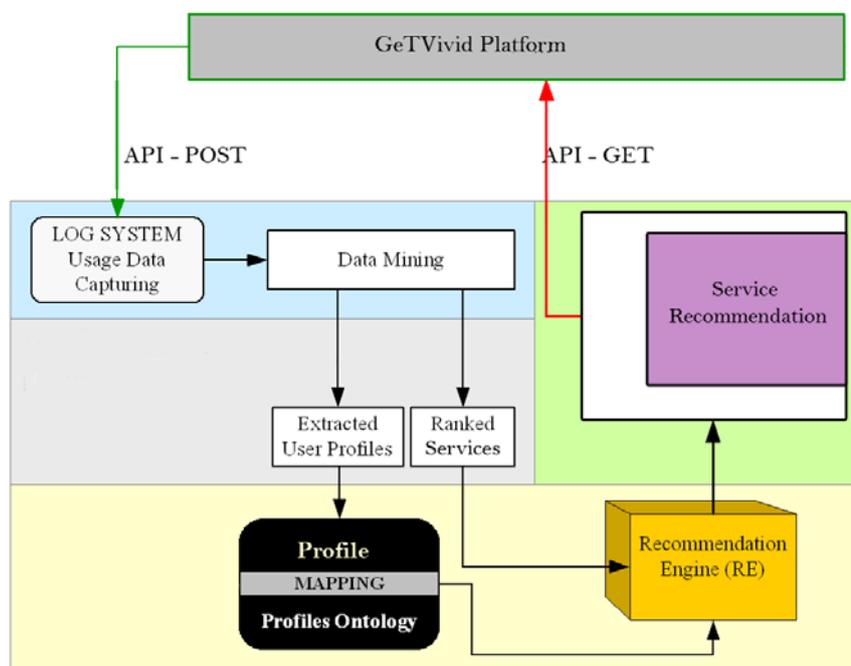


Figure 13: Profiling internal architecture

3.2.3 Data mining algorithms

Data mining algorithms are the CORE module of the profiler system, although it is still in the development and improvement, current version (V.0.2) includes the following data mining algorithms:

- Classifier Methods
 - J48 (inherit from V.0.1)
- Clustering Methods
 - Simple K-means J48 (inherit from V.0.1)
- Association Methods
 - Apriori
- K-nearest neighbourhood

For the GetVivid project, the use of data mining and profiling impact will consist of obtaining personalized services for the users, giving suggestions about related services to the more often used by the users and suggesting groups or activities to join similar to the user, or her/his preferred activities. This means a friendly environment focused on the user who will feel more comfortable with the GetVivid platform.

Since the description of the used profiling algorithms is very generic and not specific to the GetVivid project, but non the less very important for understanding the scope of the project, a detailed description is available in Annex A.

3.3 Additional functionality (PLUS)

The additional functionality covers all services that are needed in addition to the ACS. As defined above, the ACS is the central component, because it includes the user database and management. The additional functionality includes REST endpoints for profiler integration and for messaging (including group and contact management). Since there is no dedicated deliverable for this part of the system, the description below is more detailed than the description of components with dedicated deliverables (e.g. ACS). The REST endpoints introduced in this section are designed for the HbbTV and Mobile client. Furthermore, a JavaScript library to communicate with the REST services is continuously developed and updated. It is used by the HbbTV and Mobile client.

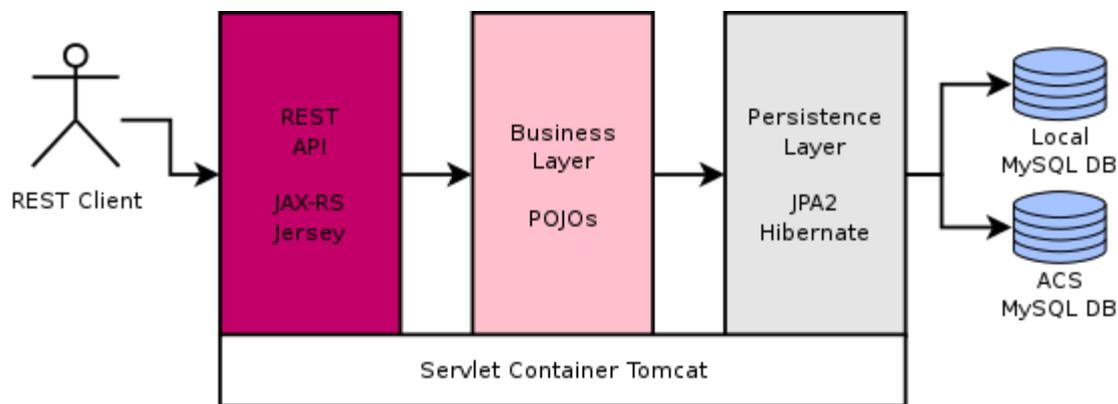


Figure 14: Layer conception of additional functionality servlet.

3.3.1 Conception

The additional functionality covers all services that are not implemented in the ACS. The corresponding requirements are defined in user stories 08.01. - 08.04. (see Annex B), namely user story 8.1 covers one-to-many chat functionality, 8.2/8.3 automatic system notifications and 8.4 contact and group management. Furthermore, the additional functionality takes care of the integration of the profiling components into the authorization scheme. Each endpoint, that requires a valid user account, validates the access token sent in a header field of the incoming HTTP request. Figure 14 shows the basic architecture of this REST service. It can be divided into REST API, Business Layer and Persistence Layer:

- The endpoints are implemented in a Tomcat7 servlet. The REST API is based on the interfaces and annotations defined in the package `javax.ws.rs`. Currently Jersey 2.* is used, which is the JAX-RS reference implementation. The access token validation and local caching is implemented as JAX-RS ResourceFilter. It can be activated by simply adding the Annotation `@Authenticated` to an endpoint definition.
- The business logic mainly contains filters and code for creation and modification of entities as well as searching and returning entities from the database.
- Hibernate was chosen as framework for the service's persistence layer. Hibernate is a popular JPA2 implementation. The persistence layer communicates with the local database as well as with the ACS remote database.

3.3.2 REST API

3.3.2.1 Endpoints related to message service

Listings 2 and 3 show a generic HTTP REST interface for searching and listing entities from the persistent data storage. The interfaces can be reused for all types of entities. The SecurityContext is automatically injected. It is created in the authentication filter and contains information associated to the given access token. This also includes the user's ID.

Anyone familiar with JAX-RS REST APIs will notice, that the interfaces define the return type String and not an object type, which would then be mapped by an object to JSON processor (e.g. Jackson). But to avoid serialization conflicts during the conversion, which would occur when mapping Hibernate DTOs to JSON strings, the conversation is done with a dedicated method in each DTO/POJO.

```
@Path("search")

public interface ISearch {

    /**
     * Search for T
     * @param search query
     * @param limit
     * @param offset
     * @param securityContext
     * @return list of T with MetaInfo
     */

    @GET
    @Produces(MediaType.APPLICATION_JSON)
    public String search(@QueryParam("q") String query,
```

```

    @QueryParam("offset") int offset, @QueryParam("limit") int limit,
    @QueryParam("sort") String sort,
    @Context SecurityContext securityContext) throws SearchQueryException;
}

```

Listing 2: Generic interface for HTTP REST search service

```

@Path("details")
public interface IDetails {
    /**
     * Details of T
     * @param idList
     * @param securityContext
     * @return list of detailed information about T with MetaInfo
     */
    @GET
    @Produces(MediaType.APPLICATION_JSON)
    public String details(@QueryParam("ids") String idList,
        @Context SecurityContext securityContext) throws DetailsException;
}

```

Listing 3: Generic interface for HTTP REST details service

Besides the requested payload the responses also contain meta information about the response. The meta information can be defined for each entity type. I usually contains the time it took server to process the request and to generate the response. In case of optional parameters it can contain the predefined defaults. The details service includes pagination offsets, page and result sizes in the meta info object.

Listing 4 shows an exemplary response by the endpoint used for searches in the user database. This endpoint is used, for example, to find new contacts for an XMPP chat.

```

GET: users/search?q=bob&offset=0&limit=10&sort=birthDate
{
  "result": [
    {
      "id": 6,
      "username": "bob",
      "firstName": "Bob",
      "lastName": "GeTVivid",
      "birthDate": "null"
    }
  ],
  "meta": {
    "limit": 10,
    "offset": 0,
    "totalResults": 1,
    "query": "bob",
    "searchTime": 0.01018286
  }
}

```

Listing 4: Example response of user search REST endpoint.

3.3.2.2 Profiler client endpoint

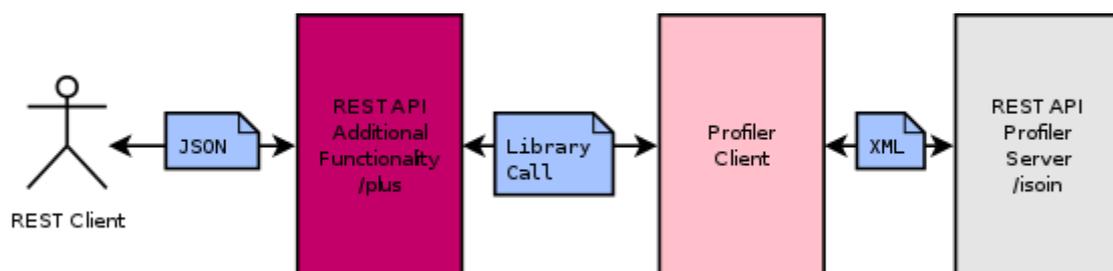


Figure 15: Architecture of profiler client endpoint.

Listings 5 to 7 show the interfaces used for the integration of the profiling component. These endpoints are used for the new offer wizard. Again, the SecurityContext is automatically injected and contains the current user's ID.

The business logic behind the profiler endpoints is pretty straight forward. A request to the profiler-server component is built with the profiler-client. Some predefined values are passed on as well as the given HTTP parameters. The response returned by the profiler-client is then converted to a POJO and returned as JSON string via the REST API. See Figure 15 for the schema of a call to the profiler.

```
public interface ISuggestCategory {  
    /**  
     * Sends a suggestion request to the profiling API.  
     * @param title  
     * @param language  
     * @param securityContext  
     * @return a category id with MetaInfo  
     * @throws ProfilerException  
     */  
    @Path("suggestCategory")  
    @GET  
    @Produces(MediaType.APPLICATION_JSON)  
    @Authenticated  
    public String suggestCategory(@QueryParam("title") String title,  
        @QueryParam("language") String language,  
        @Context SecurityContext securityContext) throws ProfilerException;  
}
```

Listing 5: Interface for the suggestion of a category based on user, language and title.

```
public interface ISuggestLocation {  
    /**  
     * Sends a suggestion request to the profiling API.  
     * @param categoryId  
     * @param language  
     * @param securityContext  
     * @return a location with MetaInfo  
     * @throws ProfilerException  
     */  
}
```

```

@Path("suggestLocation")

@GET

@Produces(MediaType.APPLICATION_JSON)

@Authenticated

public String suggestLocation(@QueryParam("categoryId") int categoryId,
    @QueryParam("language") String language,
    @Context SecurityContext securityContext) throws ProfilerException;
}

```

Listing 6: Interface for the suggestion of a location based on user, language and category identifier.

```

public interface ISuggestTime {

    /**
     * Sends a suggestion request to the profiling API.
     * @param categoryId
     * @param language
     * @param securityContext
     * @return a date/time with MetaInfo
     * @throws ProfilerException
     */

    @Path("suggestTime")

    @GET

    @Produces(MediaType.APPLICATION_JSON)

    @Authenticated

    public String suggestTime(@QueryParam("categoryId") int categoryId,
        @QueryParam("language") String language,
        @Context SecurityContext securityContext) throws ProfilerException;
}

```

Listing 7: Interface for the suggestion of a date and time based on category identifier, user and language.

3.3.3 Message service

Figure 16 shows how the XMPP server (Prosody) is integrated with the rest of the system. On the one hand this illustrates how all parts of the architecture work together. On the other hand it should be visible, that the usage of open standards, such as OAuth 2.0 or XMPP, allows integrating and replacing open source software with minimal effort. This allows fast development of new features and enables the developer to change existing features by replacing a sub component or installing resp. create a plug-in to extend the functionality of open source software.

For a detailed description of the interface between the XMPP server and the endpoints used for user probing and authentication see Listing 8. Again, both endpoints are implemented with the RESTful framework Jersey, but they do not strictly follow the usual REST conventions. For example, the response does not contain a structured object (e.g., XML or JSON).

A Prosody authentication plug-in was developed to access the defined HTTP interface. It basically only takes the user credentials or even only the user name and performs an HTTP call to the local HTTP API offered by the Tomcat7 servlet. Prosody stores user (contact list) related information in the local MySQL database, which can easily be accessed or modified by the web services running on the same machine. For example, it can be necessary to limit access to certain information to members of the user's contact list or to a certain group. Both can be accomplished by simply querying the database.

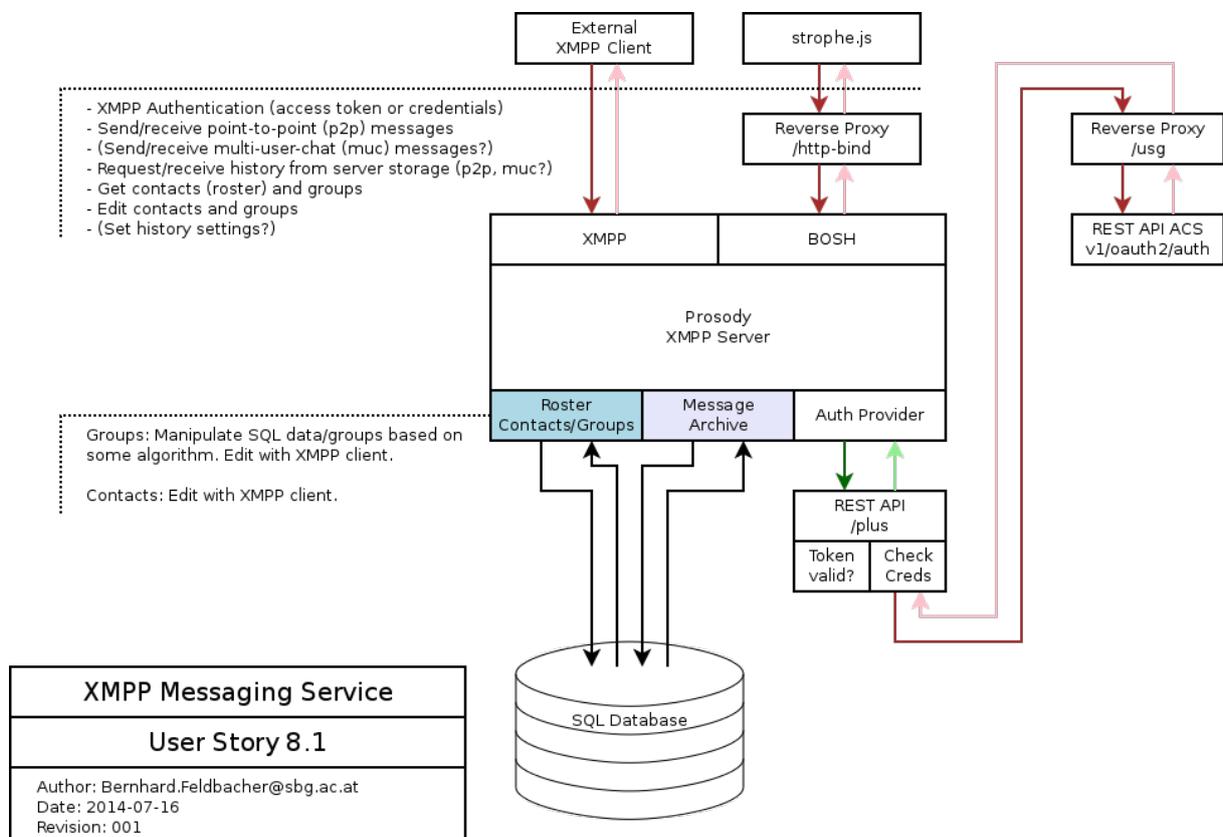


Figure 16: Integration of Prosody XMPP server

```

@Path("auth")

public interface IAuth {

    /**
     * Check credentials
     * @param username
     * @param password
     * @param realm
     * @return "true" or "false"
     */

    @POST

    @Consumes(MediaType.APPLICATION_FORM_URLENCODED)

    @Produces("text/plain")

    public String authenticate(@FormParam("username") String username,

        @FormParam("password") String password,

        @FormParam("realm") String realm);

    /**
     * Check if a user exists for a given username
     * @param username
     * @return "true"
     * @throws NotFoundException
     */

    @HEAD

    @Produces("text/plain")

    @Path("users/{username}")

    public String exists(@PathParam("username") String username)

        throws NotFoundException;

}

```

Listing 8: Interface for authentication services used by Prosody.

3.4 HbbTV client (IRT)

The following chapter shall give an overview about the architectural aspects of the TV client in the GeTVivid platform. The HbbTV specification, which defines a set of standards, adopted from the broadcast and internet world enabling interactive television, was used as a basis for the development. Also the Second-Screen-Framework, which is an extension to HbbTV applications offering intuitive user input via a mobile device, will be described.

3.4.1 Platform high level architecture

The hybrid broadcast broadband TV (HbbTV) iTV system was originally developed for the use in the DVB broadcasting system and is based on widely used web technologies. It utilises the functionality provided in existing web standards by defining sub-sets of these standards, also called profiles. Standards profiled in HbbTV include, specifications from the Open IPTV Forum (OIPF), CEA, DVB and W3C, in particular CE-HTML and JavaScript. The HbbTV specification encompasses the necessary broadcasting signal and specifies the requirement on the browser integrated in the smart TV sets. HbbTV combines data and applications received via the broadcast signal, with services, applications and content provided via the internet. Currently the next release of HbbTV, commonly referred to as HbbTV 2.0, is being specified by the HbbTV Association. It will further align HbbTV with HTML5, CSS3 and current DOM specifications, providing for an even tighter interoperability between traditional internet applications and iTV.

3.4.2 Technical aspects

HbbTV covers more than only the functionality of a browser running on a TV. The HTML pages represent the applications and have to be signalled to the terminal via the broadcast transport stream. Because some apps are bound to the linear program service, the Application Lifecycle has to be taken care of, as well as some other TV-specific issues, which were integrated in the specification (e.g., DRM, CI+ and subtitles). There's also a possibility to synchronise application events with the broadcast program (stream events).

3.4.2.1 Application signalling and receiving process

The information for managing the application is received by a hybrid broadcast broadband scheme. The broadcast interface consists of a stack of modules that demultiplex MPEG2-transportstream packages and parse the service information tables, looking for an Application Information Table (AIT). The AIT provides the information needed for application signalling and application reception either via broadband using Internet protocols or via the broadcast signal using a DSM-CC carousel (see Figure 17).

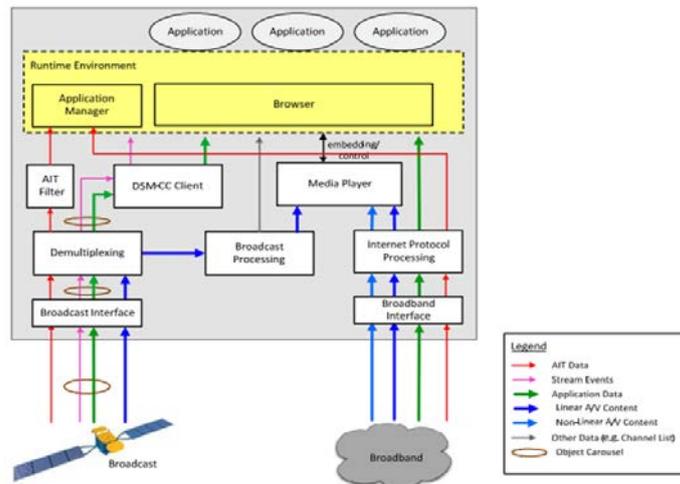


Figure 17: HbbTV architecture

Digital storage media command and control (DSM-CC) is a toolkit for developing control channels associated with MPEG-1 and MPEG-2 streams. In this context, it is used to deliver internet data via the broadcast channel. The DSM-CC allows to transmit small applications (like teletext replacement) to viewers who do not connect their devices to the internet, by minimizing web server load by transmitting a small autostart launcher application and link to web resources in a second usage step. However the conventional way of accessing HbbTV applications is by a HTTP request from a broadband-connected service. Data from both delivery channels can be processed seamlessly in one application.

3.4.2.2 Application types

Generally there are two types of applications when referring to HbbTV (also see Figure 18):

- Bound application (broadcast-related)
 - bound to the TV service and is started when the service is tuned
 - autostart application running as an overlay to the currently running broadcast service
 - usually activated in full screen mode by the red button on the remote control
 - is killed when another broadcast-service is tuned
- Unbound application (broadcast-independent)
 - accessible through a portal delivered by TV manufacturers (similar concept like app stores)
 - can be activated by other applications

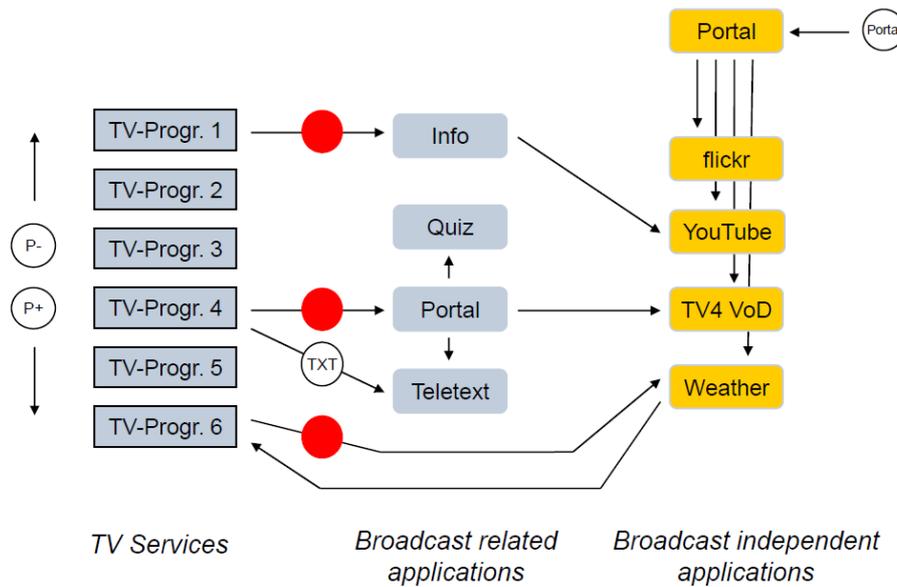


Figure 18: HbbTV architecture - application types

3.4.3 Second Screen Framework

The Second Screen Framework is tailored for HbbTV applications and allows them to be expanded, providing a bidirectional communication path with a browser based application on the second device. This allows HbbTV applications to be controlled from a second screen, or specific information to be “pushed” from the TV set on to the second screen. To create a connection between two devices, the user has to scan a QR code for a specific application into the second screen device from the TV screen. When the connection is made, the Second Screen application will start up automatically on the second device.

Since the solution is fully compliant with the HbbTV standard, it enables content providers to create fully interactive applications with direct program relation potentially targeting millions of already deployed devices on the market. Thus, the concept can be implemented without modifications to hardware and only minimal extensions to existing applications.

The Second Screen Framework provides the following features to applications:

- Discovery of devices
- Connection
- App-to-app communication
- App launch

Communication between a TV application and its companion application on the second screen is handled by a web server. Discovery between the devices is accomplished by means of a QR code. Both devices get a unique identifier from the framework server (see Figure 19). The IDs are stored in cookies in the devices’ browsers. The ID pairs are stored in a database on the framework server. Thus the connection between the devices is persistent. Users need to execute the discovery process only once. The framework is hosted as a service that is independent from the application that is making use of it.

End users can benefit from their device connections in every service that supports the framework, without having to manage application-specific connection settings. Since the app-to-app communication is done over the open Internet, there are no specific requirements regarding the network configuration to be implemented by the end-user (e.g., the devices do not need to be in the same local network). The only requirements regarding the technical set-up of the end-users devices are that both devices need an Internet connection and that they have a JavaScript-enabled web browser installed. Although the current implementation uses a centralized server, the 2.0 version of HbbTV shifts the logic of coupling and communication to the end-devices, making an intermediate server redundant in the future.

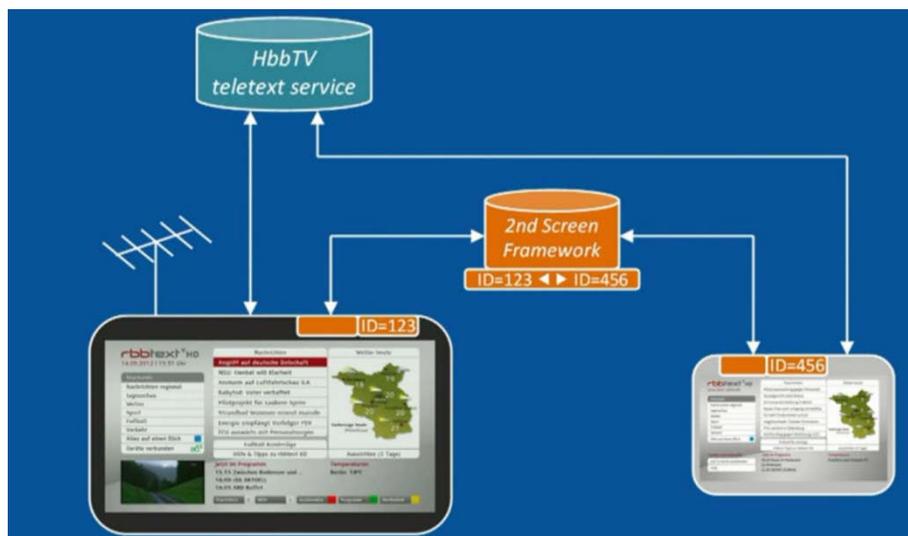


Figure 19: Second Screen Framework

3.5 Mobile client (EVISION)

In accordance to the project proposal, the Mobile-Client is meant to be a “Second-Screen-Device” for the GetVivid platform running on a HbbTV set-top box at the home of older adults. In this role it will not only be a display screen, but more the main input and controlling device for the whole system.

Due to their sophisticated hardware specifications, modern mobile devices like Smartphones and Tablets provide the most convenient way of system interaction for the user. Especially, the intuitive and reliable touch interaction promotes a maximum level of usability even for more complex controlling tasks, for younger people as well as for older adults with mild impairments.

The first round of end user evaluations has shown that tablet devices with a display size of about 10 inch suites best for the tasks of GetVivid. The devices are not too big and not too heavy, so the end user can handle them well. And the big size display together with a higher resolution provides more space for bigger icons and font-sizes, so all the functions and symbols can be recognized more easily.

For the purposes of the development and evaluation process during the project time, the client is being optimized for devices with Android operating system (V4.4.x “KitKat” or higher), because it is the most common “open” standard in mobile operating systems. But due to the “thin-client” architecture, as described

in the following section of this document, it will be compatible with other versions of the operating system. There will be also no further hardware requirements, except of a running WiFi connection. This is anyway a mandatory hardware feature for all available mobile devices on the market. Nevertheless a display size of about 10 inch is highly recommended.

3.5.1 Client conception

The Mobile-Client Application is developed as a “hybrid” application. Hybrid means that the core part of the application is basically a web-client connecting to the web-application hosted on the “static content server” (demo.getvivid.eu - see section 2.5.1), via a WiFi Internet connection. This core part is embedded in a native application, developed for a specific target platform, like mobile devices with Android operating system. While the web-application provides all functions and content for the web-client, the surrounding native app is just dealing with hardware specific functions, like input listeners and sensors, in case they are needed. This way the Mobile-Client appears - despite of its web-application core - as a “normal” App, providing the common “app-like” user experience.

Basically the Mobile-Client is free of the strict limitations of the TV Client, as they are specified in the current HbbTV V1.1 standard. Nevertheless, it sticks to some of this limitation, for example no usage of HTML5 and no other script languages rather than JavaScript. The reason for this decision is that the Mobile-Client, as a Second-Screen-Device for the TV, should have an equal appearance to the user. Furthermore, both clients need to provide a synchronized interaction. To meet this requirement it is useful to keep the conception and architecture as well as the used technologies on both clients similar. The web-application for the Mobile-Client is built with standard web-development techniques like HTML (V4.01) and CSS. Additional functions are realized in JavaScript.

This architecture concept promotes also the possibility to use the Mobile-Client as a full stand-alone-device to interact with the GetVivid platform, in case no suitable TV is available. For a more detailed description of this conception, please refer to D4.1 (Documentation of the Mobile-Client).

3.5.2 Architecture overview of the mobile client

The high-level architecture of the Mobile-Client and its most important components are shown in the following layer diagram (Figure 20).

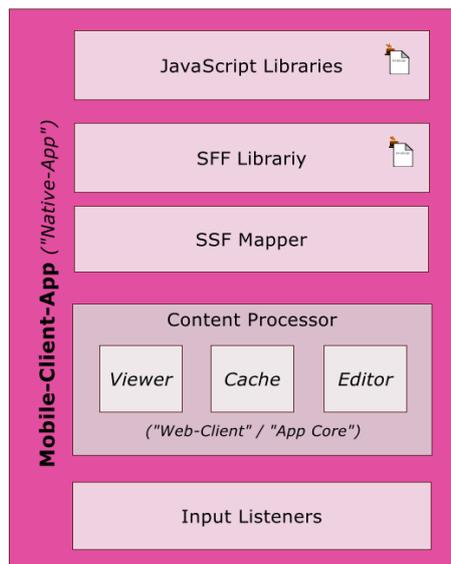


Figure 20: Layer diagram of the mobile client

The Input Listeners are hardware connected components responsible for catching all input activities on the device. Physical input activities of the user are “translated” by the operating system into so called “Events”. The corresponding Input Listener of the App is listening to this input “Events”, like for example onTouch() or onClick() and processes them according to the application logic.

The Content Processor is a browser-like component dealing with the web-content, in terms of viewing, caching and editing. It represents the “web-application core” as it is mentioned in section 3.5.1. In Android this standard component is called “android.webkit”.

The SSF Mapper is part of the synchronization process. It generates/processes synchronization messages send via the SSF to/from the TV-Client. The principal function of this component and the synchronization process is explained in section 3.4.3 as well as in section 3.5.4 and D4.1. The SSF Library provides the API necessary for the coupling and the message exchange between the clients. (see also section 3.4.3 and 3.5.4)

JavaScript libraries are used for the implementation of all advanced functions of the web-application. The “client library” developed by PLUS provides the API for the communication with the GetTVivid backend, like requesting and storing user data. But JavaScripts are also used, to create functions inside the Mobile-App itself, for example the validation of user input data (check of mandatory fields) is implemented in JavaScript.

3.5.3 System overview without synchronization

Figure 21 shows the basic system architecture first without the usage of the Second-Screen-Framework.

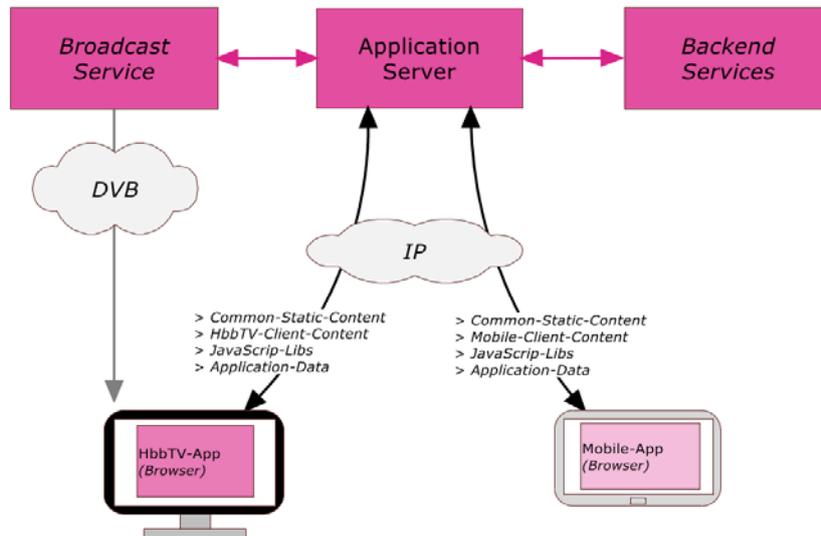


Figure 21: Basic architecture without SSF

The “GetVivid Broadcast Service” delivers in the TV stream an embedded URL as starting point of the HbbTV-Client Application. Once the application is started in full screen mode, it requests the necessary content from the “GetVivid-Application Server”. (For a detailed description, please refer to section 3.4).

The Mobile-Client is able to request the web-application content directly once the GetVivid App is started on the device. Mainly the content can be specified as:

- "Common-Static-Content" like the graphic components of the user interface, common "CSS Style Sheets" and the common "Java Script Libraries".
- In addition to this common content, used by both clients, each client has some specific content, like special style sheets and JavaScripts, dealing for example with the different ways of input and control on both devices, classic remote control versus touch interface.
- And the application data itself, like for example user input data or a catalogue of the open offers and demands posted by users.

This way, both client applications have full access to the system, but both are working “independently” from each other. It is possible to work with the system on both devices at the same time but since they are not coupled via the Second-Screen-Framework there is no synchronization between the applications.

3.5.4 System overview with synchronization

In order to establish the Mobile-Client as a real second screen remote control of the system some kind of synchronization process is necessary. Figure 22 shows the system with the implementation of the SSF as synchronization instance. The main working principal is still the same, but additionally both clients exchange messages about the status of the current application via the SSF Server.

For example: Pushing the GetVivid logo (“Home-Button”) on the mobile device navigates from any point of the application back to the starting page “index.html”. Beside this standard action, a status message like “GoTo:

index” will be generated by the SSF Mapper and send to the TV Client using the SSF Library (see section 3.5.2). After processing the message, the TV client is able to navigate to the corresponding page on the television.

This way the applications on both devices are always at the same status, displaying exactly the same user interface on their screens. The exchange and synchronization process is explained in more detail in section 3.4.3, as well as in D4.1.

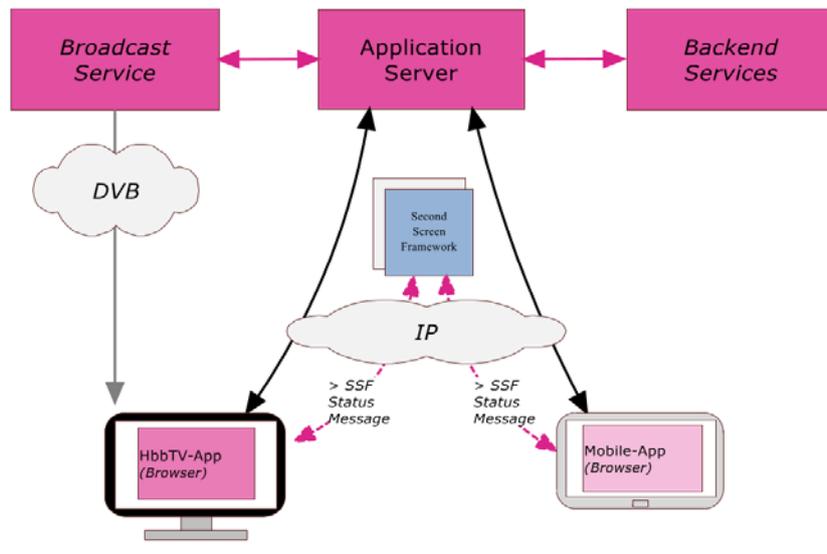


Figure 22: Architecture with SSF

4. OVERALL CONCLUSION

All details described above by the technical consortium members of the GetVivid Project must be seen as the basis for future work and development. It must be clear that the basis for the overall system is stipulated by all members. However, if alterations and modifications are necessary, the status quo can be overridden.

What is still under evaluation / in the development phase are the things beyond the pure functionality: For example, games for the older adults (including a possible aid against upcoming dementia), special features like titles and achievements as a motivation for more and efficient use of the GetVivid platform, the possibility to rate a service / a person that gives a service, etc.

All this will of course precipitate itself in alterations of the status quo system architecture and design.

REFERENCES

- A Tutorial on Clustering Algorithms. Retrieved December 5, 2014 from http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/.
- Alexander, D. 2006. Data Mining. (March 2014). Retrieved December 5, 2014 from <http://www.eco.utexas.edu/~norman/BUS.FOR/course.mat/Alex/>.
- Arnold, S.E. 2001. *Internet users at risk: the identity/privacy target zone*. Searcher. 9(1), 24-39.
- Association Rule Learning. (November 2014). Retrieved December 5, 2014 from http://en.wikipedia.org/wiki/Association_rule_learning.
- Birrer, F. A. J. 2005. *Data mining to combat terrorism and the roots of privacy concerns*. Ethics and Information Technology, 7(4), 211-220.
- Cluster Analysis. (November 2014). Retrieved December 5, 2014 from http://en.wikipedia.org/wiki/Cluster_analysis.
- Cummins, C. 2006. *Below the Surface*. Library Journal, Winter 2006, 12-14.
- Data Mining Algorithms. (April 2014). Retrieved December 5, 2014 from <http://msdn.microsoft.com/en-us/library/ms175595.aspx>.
- K-nearest neighbors. (November 2014). Retrieved December 5, 2014 from http://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm.
- Lipowicz, A. 2006. Washington Technology; *Data Mining Gets a Makeover: Call it Fusion as New Tools Expand Hunt for Terrorists*. (FEDERAL) Washingtonpost Newsweek Interactive.
- Saed, S. 2014. An Introduction to Data Mining. (November 2014). Retrieved December 5, 2014 from http://www.saedsayad.com/k_nearest_neighbors.htm.
- Schiaffino, S. and Amandi A. 2009. *Intelligent User Profiling*. Springer, Berlin, Germany.
- Solove, D. 2004. *The digital person: technology and privacy in the information age*. NYU Press.
- Tan, P.N. and Steinbach, M. and Kumar, Vipin 2006. *Introduction to Data Mining*. Addison-Wesley, Boston, MA.

ANNEX A

This Annex was created to explain the classification methods mentioned in section 3.2.3 to facilitate the reading of the general platform technical overview.

Classification methods

First version is focused on the J48 tree decision algorithm which is an implementation of the C4.5, used to classify a data model or a single instance, based on a previous trained model.

C4.5 builds decision trees from a set of training data in the same way as ID3, using the concept of information entropy. The training data is a set $S = \{s_1, s_2, \dots\}$ of already classified samples. Each sample s_i consists of a p -dimensional vector $(x_{1,i}, x_{2,i}, \dots, x_{p,i})$, where the x_j represent attributes or features of the sample, as well as the class in which s_i falls.

At each node of the tree, C4.5 chooses the attribute of the data that most effectively splits its set of samples into subsets enriched in one class or the other. The splitting criterion is the normalized information gain (difference in entropy). The attribute with the highest normalized information gain is chosen to make the decision. The C4.5 algorithm then recurs on the smaller sublists.

This algorithm has a few base cases. All the samples in the list belong to the same class. When this happens, it simply creates a leaf node for the decision tree saying to choose that class:

- None of the features provide any information gain. In this case, C4.5 creates a decision node higher up the tree using the expected value of the class.
- Instance of previously-unseen class encountered. Again, C4.5 creates a decision node higher up the tree using the expected value.
- Imagine that you have a dataset with a list of predictors or independent variables and a list of targets or dependent variables. Then, by applying a decision tree like J48 on that dataset would allow you to predict the target variable of a new dataset record.

Clustering

Clustering can be considered the most important unsupervised learning problem; so, as every other problem of this kind, it deals with finding a structure in a collection of unlabeled data.

A loose definition of clustering could be “the process of organizing objects into groups whose members are similar in some way”. A cluster is therefore a collection of objects which are “similar” between them and are “dissimilar” to the objects belonging to other clusters. We can show this with a simple graphical example (see Figure 14).

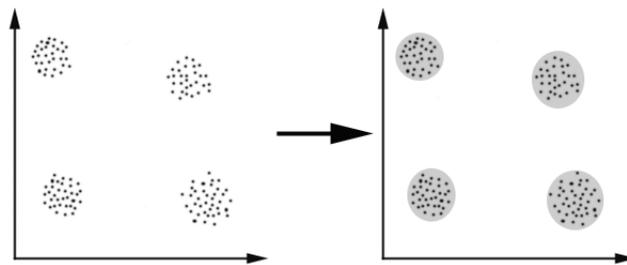


Figure 23: Clustering

In this case the 4 clusters into which the data can be divided are easily identified; the similarity criterion is distance: two or more objects belong to the same cluster if they are “close” according to a given distance (in this case geometrical distance). This is called distance-based clustering.

Another kind of clustering is conceptual clustering: two or more objects belong to the same cluster if this one defines a concept common to all that objects. In other words, objects are grouped according to their fit to descriptive concepts, not according to simple similarity measures.

Other utilities: how many clusters, instances of the clusters,...

K-means

First version of the clustering algorithms is focused on the K-means, this algorithm is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori. The main idea is to define k centroids, one for each cluster. These centroids should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and an early groupage is done. At this point it is necessary to re-calculate k new centroids as barycenters of the clusters resulting from the previous step. After these k new centroids are calculated, a new binding has to be done between the same data set points and the nearest new centroid. A loop has been generated. As a result of this loop one may notice that the k centroids change their location step by step until no more changes are done. In other words centroids do not move any more.

Finally, this algorithm aims at minimizing an objective function, in this case a squared error function. The objective function

$$J = \sum_{j=1}^k \sum_{i=1}^n \|x_i^{(j)} - c_j\|^2$$

Where $\|x_i^{(j)} - c_j\|^2$ is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster centre c_j , is an indicator of the distance of the n data points from their respective cluster centres.

The algorithm is composed of the following steps:

1. Place K points into the space represented by the objects that are being clustered. These points represent initial group centroids.
2. Assign each object to the group that has the closest centroid.
3. When all objects have been assigned, recalculate the positions of the K centroids.
4. Repeat Steps 2 and 3 until the centroids no longer move. This produces a separation of the objects into groups from which the metric to be minimized can be calculated.

Association methods consist on giving of transactions and finding rules that will predict the occurrence of an item based on the occurrences of other items in the transaction. Association is useful for discovering interesting relationships hidden in large data set, the uncovered relationships can be represented in the form of association rules or sets of frequent items, as can be seen in the example (see Listing 2).

Market-Basket transactions

1 Bread, Milk

2 Bread, Diaper, Beer, Eggs

3 Milk, Diaper, Beer, Coke

4 Bread, Milk, Diaper, Beer

5 Bread, Milk, Diaper, Coke

Example of Association Rules

$\{\text{Diaper}\} \rightarrow \{\text{Beer}\},$

$\{\text{Milk, Bread}\} \rightarrow \{\text{Eggs, Coke}\},$

$\{\text{Beer, Bread}\} \rightarrow \{\text{Milk}\},$

Implication means co-occurrence,
not causality!

Listing 9: Association rules

K-nearest neighbourhood

K nearest neighbourhood is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure (e.g., distance functions). KNN has been used in statistical estimation and pattern recognition since the beginning of 1970's as a non-parametric technique.

A case is classified by a majority vote of its neighbours, with the case being assigned to the class most common amongst its K nearest neighbours measured by a distance function (see Figure 16). If $K = 1$, then the case is simply assigned to the class of its nearest neighbour.

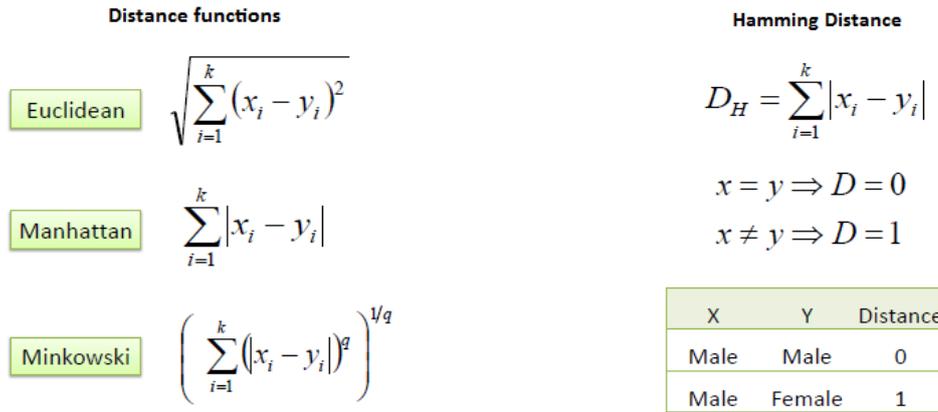


Figure 24: Distance functions

It should also be noted that all three distance measures are only valid for continuous variables. In the instance of categorical variables the Hamming distance must be used. It also brings up the issue of standardization of the numerical variables between 0 and 1 when there is a mixture of numerical and categorical variables in the dataset.

Choosing the optimal value for K is best done by first inspecting the data. In general, a large K value is more precise as it reduces the overall noise but there is no guarantee. Cross-validation is another way to retrospectively determine a good K value by using an independent dataset to validate the K value. Historically, the optimal K for most datasets has been between 3-10. That produces much better results than 1NN.

Example:

Consider the following data concerning credit default. Age and Loan are two numerical variables (predictors) and Default is the target (see Figure 16).

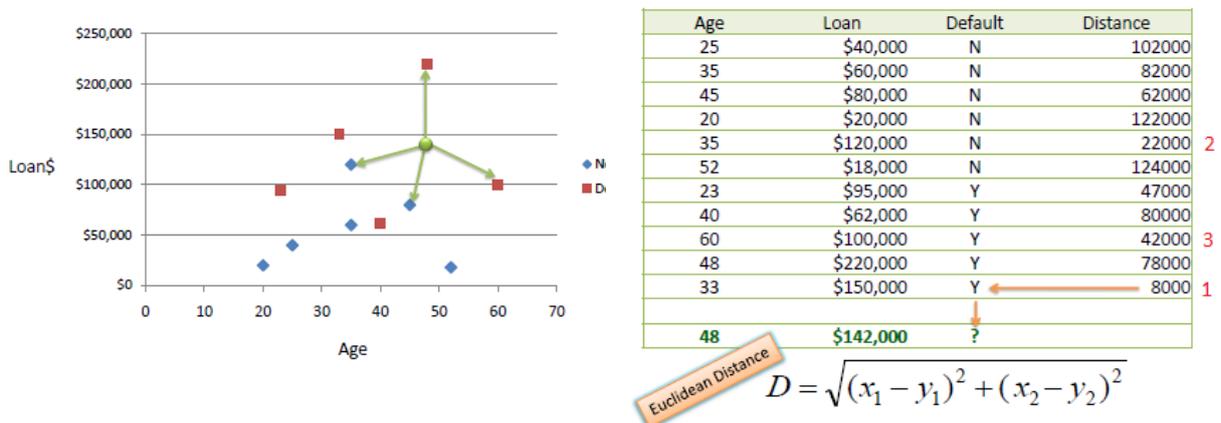


Figure 25: Example concerning credits

We can now use the training set to classify an unknown case (Age=48 and Loan=\$142,000) using Euclidean distance. If $K=1$ then the nearest neighbor is the last case in the training set with Default=Y.

$$D = \text{Sqrt}[(48-33)^2 + (142000-150000)^2] = 8000.01 \gg \text{Default}=Y$$

With $K=3$, there are two Default=Y and one Default=N out of three closest neighbors. The prediction for the unknown case is again Default=Y.

Profiling applying data mining is an increasingly common technique that gives added value to applications. Allowing applications to give personalized services in very different scenarios or scopes, is used by governments and corporations to predict and establish trends with specific purposes in mind [Birrner, 2005; Alexander, 2006]. For example, The TIA and ADVISE use data mining is used as an anti-terrorist measure by looking for specific data to identify the terrorists before a terrorist attack [Solove, 2004; Birrner, 2005, Lipowicz, 2006] or Amazon.com uses it to promote its sales by pre-selecting items (Arnold, 2001) while Libraries use data mining to become more efficient in developing their collections and management of staff [Cummins, 2006].

ANNEX B - USER STORIES

Wikipedia defines user stories as follows: In software development and product management, a **user story** is one or more sentences in the everyday or business language of the end user or user of a system that captures what a user does or needs to do as part of his or her job function. User stories are used with agile software development methodologies as the basis for defining the functions a business system must provide, and to facilitate requirements management. It captures the 'who', 'what' and 'why' of a requirement in a simple, concise way, often limited in detail by what can be hand-written on a small paper notecard.

User stories are written by or for business users or customers as a primary way to influence the functionality of the system being developed. User stories may also be written by developers to express non-functional requirements (security, performance, quality, etc.), though primarily it is the task of a product manager to ensure user stories are captured.

When the time comes for creating user stories, one of the developers gets together with the product owner (in Scrum), which has the responsibility for formulating the user stories. The developer may use a series of questions to get the customer representative going, such as asking about the desirability of some particular functionality, but must take care not to dominate the idea-creation process.

As the customer representative conceives a user story, it is written down on a note card (here a confluence (sub-) page was created) with a name and a brief description. Commonly, user stories are not to be definite once they have been written down, since requirements tend to change throughout the development lifecycle, which agile processes handles by not carving them in stone upfront.

(Source: http://en.wikipedia.org/wiki/User_story)

ID	Item
Platform Monitoring and Controlling	
01.01	Registration Approval As a user [administrator] of the GeTVivid system I want the registration to be conducted So that a registration process can be completed automatically
01.02	New Professional Offer Approval As a user [administrator] of the GeTVivid system I want to approve the first professional offer of each user [provider] So that I can sort out abusive services.
01.03	<u>Offer or Demand Rejection</u> As a user [administrator] of the GeTVivid system I want to reject n offer or demand of a user [provider, consumer] So that I can remove abusive services.
01.04	User Behavior Management As a user [administrator] of the GeTVivid system I want to ban an abusive user So that his or her account is deactivated until the matter is solved
01.05	Report a User

	As a user [consumer, provider] of the GetVivid system I want be able to report another user [consumer, provider] (e.g. abusive usage of the GetVivid system). So that the misbehavior can reviewed and verified and the necessary actions (warning, sanctions, deletion, possible other actions?) be executed.
01.06	Automatic Fraud Prevention Profiling, Anomalies
Registration Process	
02.01	User Driven Registration As a user [prospective consumer, prospective provider] of the GetVivid system I want to create an user account for myself So that I can use the full range of functions
02.02	Administrator driven registration As a user [platform administrator] of the GetVivid system I want to create a user account for another user [prospective consumer, prospective provider] So that the user [prospective consumer, prospective provider] is able to use the full range of functions
02.03	Consumer driven activation As a user [consumer] with an account that has just been registered I want to receive an eMail containing an activation link So that I can activate my account by clicking on the link or by copy/paste the link into an online form
02.04	Administrator driven activation As a user [platform administrator] I want to browse a list of accounts, that have not been activated yet So that I can activate his / her account by clicking on an activation button next to the user name/identification.
Login and Logout	
03.01	Login As a user [consumer] of the GetVivid system I want to log into the system So that I am recognized by the system, connected to the system and have a permanent connection with the system
03.02	Logout As a user [consumer] of the GetVivid system with an active log in I want to log off the system So that I am no longer recognized by the system and I can log on again with the same or a different account.
Device Coupling and Decoupling	
04.01	Coupling TV and Tablet As a user [consumer] with an activated GetVivid account I want to couple a HbbTV device with a tablet computer by photographing a QR-code from the TV or by typing in a displayed TAN So that I can use the tablet as a second screen device for displaying information and navigation
04.02	Decoupling TV and Tablet As a user [consumer] with two coupled devices I want to decouple my devices So that I can couple them with other devices.

Offer-driven approach	
05.01	<p>Posting a General Offer - Professional</p> <p>As a user [provider] with an activated account to which I am currently logged-in to</p> <p>I want to offer my services to other users [consumer] by adding them to the catalog via the web-interface</p> <p>So that other users [consumer] can request them in an easy way at anytime</p>
05.02	<p>Responding to a General Offer – Professional</p> <p>As an user [consumer] with an activated account to which I am currently logged-in to</p> <p>I want to respond to a general offer</p> <p>So that I get a professional service</p>
05.03	<p>Posting a General Offer – Informal</p> <p>As a user [consumer] with an activated account to which I am currently logged-in to</p> <p>I want to offer my (informal) help to other users [consumer] by adding an element to the catalogue</p> <p>So that other users [consumer] can request them in an easy way at anytime</p>
05.04	<p>Responding to a General Offer – Informal</p> <p>As an user [consumer] with an activated account to which I am currently logged-in to</p> <p>I want to respond to a general offer (Informal)</p> <p>So that I can obtain the offer</p>
05.05	<p>Posting a Short-Term Offer</p> <p>As a user [consumer] with an activated account to which I am currently logged-in to</p> <p>I want to offer short-dated services and information to a defined or undefined group of other users [consumer] by sending a message to the platform</p> <p>So that the other users [consumer] can be notified on condition that they are available to the system during the period of my message-availability</p>
05.06	<p>Responding to a Short-Term Offer</p> <p>As an user [consumer] with an activated account to which I am currently logged-in to</p> <p>I want to respond to a short-term offer (Informal)</p> <p>So that I can obtain the offer</p>
05.07	<p>Posting a Joint Activity Offer</p> <p>As a user [provider, consumer] with an activated account to which I am currently logged-in to</p> <p>I want to offer joint activities in terms of multi-consumer and / or multi-variant appointments</p> <p>So that other users [consumer] can vote / subscribe for 0..n suggested appointments</p>
05.08	<p>Responding to a Joint Activity Offer</p> <p>As a user [consumer] with an activated account to which I am currently logged-in to</p> <p>I want to respond to a Joint Activity Offer</p> <p>So that I can obtain the offer</p>
05.09	<p>Modification of Offers</p> <p>As a user [consumer] with an activated account to which I am currently logged-in to</p> <p>I want to update or delete my offer</p> <p>So that it meets the actual necessities</p>
05.10	<p>System Generated (Joint) Activity Offers</p> <p>As a user [administrator]</p> <p>I want to generate Joint Activity Offers in regions with low or no activity</p> <p>So that the activity in the system is augmented.</p>
05.11	<p>Modification of an Agreement</p> <p>As a user [consumer, provider] with an activated account to which I am currently logged-in to</p> <p>I want to renegotiate an existing agreement</p> <p>So that it meets the actual necessities.</p>

Demand-driven approach	
06.01	<p>Posting a General Demand</p> <p>As a user [consumer] with an existing account to which I am currently logged-in to I want to post a certain demand in terms of a service that I am generally looking for So that other users [consumer, provider] can respond to my request by offering their help</p>
06.02	<p>Responding to a General Demand</p> <p>See 02 - Responding to a General Offer – Professional and 04 - Responding to a General Offer – Informal.</p>
06.03	<p>Posting a Short-Term Demand</p> <p>As a user [consumer] with an existing account to which I am currently logged-in to I want to post a certain demand in terms of a service that I am looking for So that other users [consumer, provider] who have marked during the configuration that they are generally willing to help in such cases are notified and can respond to my request by offering their help</p>
06.04	<p>Responding to a Short-Term Demand</p> <p>See 06 - Responding to a Short-Term Offer.</p>
User Accounts	
07.01	<p>Account Status</p> <p>As a user [consumer] with an existing account to which I am currently logged-in to I want to set the status of my account inactive or active So that I don't receive any more notifications (in order to avoid Logout/Login)</p>
07.02	<p>Configure Account for Demand Request Distribution</p> <p>As a user [consumer] with an existing account to which I am currently logged-in to I want to mark a selection of categories [provider] or a selection of category-user tuples [consumer] So that demand driven requests can be forwarded to me or notifications can be displayed in my account</p>
07.03	<p>Role Achievements</p> <p>Profiling, Gamification, User Roles (e.g. "the mechanic", "the transporter", ...)</p>
Message Service	
08.01	<p>One-To-Many (Mailinglist)</p> <p>As a user [consumer] with an existing account to which I am currently logged-in to I want to send a message to a group of users [consumer] So that those are informed</p>
08.02	<p>Automatic Notifications</p> <p>As a user [consumer] with an existing account to which I am currently logged-in to I want to receive automatic notifications So that I am able to keep track of interesting topics, my medication, the weather etc.</p>
08.03	<p>New Offer / Demand Bubble</p> <p>As a user [consumer] with an existing account to which I am currently logged-in to I want to have an indicator on the start page So that I am informed if there are any new demands or offers.</p>
Calendar	
09.01	<p>Create a Calendar Entry</p> <p>As a user [consumer] with an existing account to which I am currently logged-in to I want to create a new appointment or to-do in my personal calendar So that it helps me to organize my agenda.</p>
09.02	<p>Modify a Calendar Entry</p>

	As a user [consumer] with an existing account to which I am currently logged-in to I want to modify or delete an existing appointment or to-do in my personal calendar So that it meets the actual necessities.
09.03	Confirm a Reminder As a user [consumer] with an existing account to which I am currently logged-in to I want to be reminded at the specified time So that I don't forget an appointment or to-do.
Interaction Modalities	
10.00	Interaction between/on mobile client and HbbTV Client
User Profiling	
11.01	Suggested services based on the navigation history As a user [consumer] of the GetVivid system I want to see suggestions based on my navigation history So that I can select different services quickly
11.02	Suggested services based on purchase history As a user [consumer] of the GetVivid system I want to see suggestions based on my purchase history So that I can select different services quickly
11.03	Suggested Services based on other users As a user [consumer] of the GetVivid system I want to see suggestions based on the services which other users have been looking for So that I can select different services quickly
11.04	Suggested Services based on other users purchases As a user [consumer] of the GetVivid system I want to see suggestions based on the services which other users have hired So that I can select different services quickly
11.05	Suggest most purchased services As a user [consumer] of the GetVivid system I want to see most purchased services suggestions So that I can select different services quickly
11.06	Identity impersonation system As a user [consumer] of the GetVivid system I want to know if someone is impersonating my account So that I can take security measures
11.07	Terms abusive detector system As a user [consumer] of the GetVivid system I want to notify when I see an abusive terms service So that the system will notify the abusive terms service to the user [consumer] and to the administrator of the platform
11.08	Suggest services most popular in the user's [customer] area As a user [consumer] of the GetVivid system I want to know most popular services in my area So that I can select a service quickly

Status as of September 2014

01 Platform Monitoring and Control

01 - Registration Approval

Created by Manuel Eisele, last modified by Christoph Ortner on 12.03.2014

User Story

Registration Approval					
No		Priority	1	Size	tbd
HbbTV		Tablet		Web	
Story Narrative					
As a	user [administrator] of the GeTVivid system				
I want	the registration to be conducted				
So that	a registration process can be completed automatically				
Acceptance Criterion 1					
Given	that a user [consumer] needs approval after registration				
When	the registration process is completed and the manual approval is activated				
Then	the user [administrator] has to activate the user [consumer]				
Acceptance Criterion 2					
Given	that a user [provider] needs approval after registration				
When					
Then	the user [provider] account is activated automatically				

02 - New Professional Offer Approval

Created by Manuel Eisele, last modified by Bernhard Feldbacher on 20.03.2014

User Story

New Professional Offer Approval					
No		Priority	1	Size	tbd
HbbTV		Tablet		Web	X
Story Narrative					
As a	user [administrator] of the GeTVivid system				
I want	to approve the first professional offer of each user [provider]				
So that	I can sort out abusive services.				
Acceptance Criterion 1					
Given	that a user [provider] has posted a professional offer and the user [administrator] has not approved any offers by this user [provider] yet				
When	the user [administrator] is approving the offer				
Then	the user [provider] should be considered trustworthy and the offer should be approved / visible to the other users [consumers].				
Acceptance Criterion 2					
Given	that a user [provider] has posted a professional offer and the user [administrator] has already approved an offer by this user [provider]				
When	the offer is posted by the user [provider]				
Then	the offer should be approved / visible to the other users [consumers].				

03 - Offer or Demand Rejection

Created by Bernhard Feldbacher, last modified on 02.07.2014

User Story

Offer or Demand Rejection					
No		Priority	1	Size	td
HbbTV		Tablet		Web	X
Story Narrative					
As a	user [administrator] of the GeTVivid system				
I want	to reject an offer or demand of a user [provider, consumer]				
So that	I can remove abusive services.				
Acceptance Criterion 1					
Given	that a user [provider] has posted a professional offer and the user [administrator] has already approved an offer by this user [provider] and the service violates any of the platform guidelines (e.g. ethical)				
When	the offer or any other offer is posted by the user [provider]				
Then	the user [administrator] should be able to remove the offer or any other offer from the catalogue.				
Acceptance Criterion 2					
Given	that a user [consumer] has posted an informal offer or demand and the service violates any of the platform guidelines (e.g. ethical)				
When	the offer or demand is posted by the user [consumer]				
Then	the user [administrator] should be able to remove the offer/demand from the catalogue.				

Message Content

Comments

PLUS: Use a bad word filter, some kind of blacklist that is applied to the user input? Main goal: support the administrator ISOIN is working on text detection patterns. Progress update as soon as the user story becomes part of a sprint.

04 - User Behavior Management

Created by Bernhard Feldbacher, last modified on 18.03.2014

User Story

User Behavior Management					
No		Priority	1	Size	td
HbbTV		Tablet		Web	X
Story Narrative					
As a	user [administrator] of the GeTVivid system				
I want	to ban an abusive user				
So that	his or her account is deactivated until the matter is solved				
Acceptance Criterion 1					
Given	that the account is deactivated				
When	the matter is resolved in a positive way				
Then	the account shall be reactivated.				
Acceptance Criterion 2					
Given	that the account is deactivated				
When	the matter cannot be solved in a positive way				
Then	the account shall be deleted and the email address shall be black listed.				

05 - Report a User

Created by Christoph Ortner, last modified by Bernhard Feldbacher on 18.03.2014

User Story

Report a User					
No		Priority	1	Size	tbd
HbbTV		Tablet	X	Web	X
Story Narrative					
As a	user [consumer, provider] of the GeTVivid system				
I want	be able to report another user [consumer, provider] (e.g. abusive usage of the GeTVivid system).				
So that	the misbehavior can reviewed and verified and the necessary actions (warning, sanctions, deletion, possible other actions?) be executed.				
Acceptance Criterion 1					
Given	the user has been reported				
When	the misbehavior is verified				
Then	necessary steps shall be taken by the user [administrator].				

06 - Automatic Fraud Prevention

Created by Christoph Ortner, last modified by Bernhard Feldbacher on 02.07.2014

Comments

PLUS: Is there any way to detect anomalies in the usage of the system?

ISOIN: We have to be able to log everything that the user is able to do. (Every action will be linked with the profiling API.)

PLUS: This User Story will probably evolve with every feature we add to the system.

ISOIN: Currently we can access data from the ACS/Amonia (user agreements, searches, entries categories, entries services).

See [11.06 - Identity impersonation system](#)

2 Comments



Bernhard Feldbacher

Maybe [@Alex Garcia](#)/ISOIN has any ideas how this can be accomplished in a effective way?

The idea is to prevent abuse by persons who have access to GeTVivid system via a user's [consumer] account (i.e. home care, cleaning personal, nephew tricks etc.).

Reply • Delete • 18.03.2014



Alex Garcia

In this case profiling could help to securize the system, having a different behaviour could raise an alarm to the user email, administrator, relatives,...

Reply • Delete • 18.03.2014

07 - Administrative System Log

Created by Bernhard Feldbacher, last modified on 16.07.2014

User Story

Administrative System Log					
No		Priority	1	Size	tbd
HbbTV		Tablet		Web	X
Story Narrative					
As a	user [administrator] of the GeTVivid system				
I want	to have an overview over the activity in the system				
So that	I am able to take actions if necessary.				
Acceptance Criterion 1					
Given	a user [consumer] is offline for a certain period of time				
When	the system automatically checks the time of the last activity				
Then	I want to see an entry in the system log with the time of the last activity and the corresponding user.				
Acceptance Criterion 2					
Given	two devices belonging to a user [consumer] are coupled/decoupled multiple times				
When	they are coupled				
Then	I want to see an entry in the system log with the corresponding user.				
Acceptance Criterion 3					
Given	the fraud prevention component detects an anomaly				
When	it does an automatic check				
Then	I want to see an entry in the system log with the corresponding user and the kind of anomaly.				
Acceptance Criterion 4					
Given	a demand stays unanswered for a certain period of time				
When	the system automatically checks the database				
Then	I want to see an entry in the system log with the corresponding user and demand.				

02 Registration Process

01 - User Driven Registration

Created by Manuel Eisele, last modified by Christoph Ortner on 19.03.2014

User Story

User Driven Registration					
No		Priority	1	Size	td
HbbTV		Tablet		Web	X
Story Narrative					
As a	user [prospective consumer, prospective provider] of the GetVivid system				
I want	to create an user account for myself				
So that	I can use the full range of functions				
Acceptance Criterion 1					
Given	that all mandatory fields have been filled and all fields are validated				
When	the user [prospective consumer, prospective provider] submits the registration form				
Then	generate an activation code and send the code as a link to the user [prospective consumer or provider] in an eMail				
Acceptance Criterion 2					
Given	that the user [platform administrator] has defined, that all registration requests have to be approved by the user [platform administrator]				
When	a user [prospective consumer, prospective provider] submits the registration form				
Then	acceptance criterion 1 should be paused until the user [platform administrator] has accepted the request				
Acceptance Criterion 3					
Given	that the eMail address is already know to the system				
When	the user [prospective consumer, prospective provider] submits the registration form				
Then	a password reset option should be offered and the registration should be refused until an unknown eMail address is submitted				

Message Content

Name:	m	First name, family name
Photo/Logo:	m	Foto (Headshot) of the User, Logo of Company
eMail:	m	eMail address, the account should be connected with, e.g. for activation and notification purposes
Address:	m	Home-address of the account user resp. address of the company
Phone:	[m]	Home- and/or Mobile Number
Password:	m	For the account usage
Company name:	[m]	Mandatory for provider accounts
Company imprint:	[m]	Mandatory for provider accounts, e.g. for Germany/Austria: uid/"USt-IdNr"

02 - Administrator driven registration

Created by Manuel Eisele, last modified by Bernhard Feldbacher on 17.03.2014

User Story

Administrator driven registration					
No		Priority	1	Size	tbd
HbbTV		Tablet		Web	X
Story Narrative					
As a	user [platform administrator] of the GeTVivid system				
I want	to create a user account for another user [prospective consumer, prospective provider]				
So that	the user [prospective consumer, prospective provider] is able to use the full range of functions				
Acceptance Criterion 1					
Given	that the eMail address is already know to the system				
When	the user [platform administrator] submits the registration form				
Then	the registration is refused until an unknown eMail address is submitted				
Acceptance Criterion 2					
Given	that all mandatory fields have been filled and all fields are validated and the user [administrator] has defined the activation type				
When	the user [platform administrator] submits the registration form on behalf of the other user [prospective consumer, prospective provider]				
Then	generate an activation code and send the code as a link to the user [prospective consumer or provider] in an eMail or active the account directly without any activation code				

Message Content

Name:	m	First name, family name
Photo/Logo:	m	Photo of User, Logo of Company
eMail:	m	eMail address, the account should be connected with, e.g. for activation and notification purposes
Address:	m	Home-address of the account user resp. address of the company
Phone:	[m]	Home- or Mobile Phone Number
Password:	m	For the account usage
Company name:	[m]	Mandatory for provider accounts
Company imprint:	[m]	Mandatory for provider accounts, e.g. for Germany/Austria: uid/"UST-IdNr"

03 - Consumer driven activation

Created by Manuel Eisele, last modified by Christoph Ortner on 19.03.2014

User Story

Consumer driven activation					
No		Priority	1	Size	tdb
HbbTV		Tablet		Web	X
Story Narrative					
As a	user [consumer] with an account that has just been registered				
I want	to receive an eMail containing an activation link				
So that	I can activate my account by clicking on the link or by copy/paste the link into an online form				
Acceptance Criterion 1					
Given	that the activation code is known to the system and the account is not already activated				
When	the user [consumer] submits the code				
Then	mark his / her account as activated				

Message Content

: Activation Code m Activation code that has been sent to the user after the registration process

04 - Administrator driven activation

Created by Manuel Eisele, last modified by Bernhard Feldbacher on 27.02.2014

User Story

Administrator driven activation					
No		Priority	1	Size	td
HbbTV		Tablet		Web	X
Story Narrative					
As a	user [platform administrator]				
I want	to browse a list of accounts, that have not been activated yet				
So that	I can activate his / her account by clicking on an activation button next to the user name/identification.				
Acceptance Criterion 1					
Given	that the user [platform administrator] has confirmed the correctness of the users [consumer] eMail address again				
When	the user [platform administrator] clicks the activation button				
Then	mark the users [consumer] account as activated and send a notification via eMail to the user [consumer].				

03 Login and Logout

01 - Login

Created by Manuel Eisele, last modified by Bernhard Feldbacher on 19.03.2014

User Story

Login					
No		Priority	1	Size	td
HbbTV	X	Tablet	X	Web	X
Story Narrative					
As a	user [consumer] of the GeTVivid system				
I want	to log into the system				
So that	I am recognized by the system, connected to the system and have a permanent connection with the system				
Acceptance Criterion 1					
Given	the 2nd Screen is logged into the GeTVivid system				
When	the user [consumer] links the TV set and the 2nd Screen via the SSF				
Then	the TV set should automatically be logged into the GeTVivid system				
Acceptance Criterion 2					
Given	the TV set is logged into the GeTVivid system				
When	the user [consumer] links the TV set and the 2nd Screen via the SSF				
Then	the 2nd Screen should automatically be logged into the GeTVivid system				
Acceptance Criterion 3					
Given	the user [consumer] logs into the GeTVivid system on the mobile client manually				
When	the user [consumer] enters a valid username and password				
Then	the user [consumer] should be logged into the GeTVivid system on the current device and on any TV client connected via the SSF				

Message Content

In case of a manual log in:

User Identification: m An identification for an already registered and activated account

Password: m The valid password corresponding to the User Identification.

Comments

Login via QR-Code or via link in Browser the APP is downloaded on the 2nd Screen automatically and via another link pass the login credentials to the APP [?](#)

02 - Logout

Created by Manuel Eisele, last modified by Bernhard Feldbacher on 02.07.2014

User Story

Logout					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	X
Story Narrative					
As a	user [consumer] of the GeTVivid system with an active log in				
I want	to log off the system				
So that	I am no longer recognized by the system and I can log on again with the same or a different account.				
Acceptance Criterion 1					
Given	the tv and/or mobile client have an active log in				
When	the user [consumer] logs into the web interface				
Then	the user [consumer] can terminate the session on both devices, the coupling between the two devices is separated automatically.				
Acceptance Criterion 2					
Given	the web client has an active log in				
When	the user [consumer] clicks on the log out button				
Then	the session should be terminated and the user [consumer] should be redirected to the start page of the web client.				

Message Content

Comments

The interaction elements on TV and mobile client for the log out action should be hidden very well. The possibility should not exist to "accidentally" perform a log out.

PLUS: If the system detects an anomaly in the user's behavior, the system should probably ask for the password. At least on the Mobile Client.

ISOIN: Other ways of checking the users identity? A secret question? Connection form the same place as usual (IP address or geo location)?

04 Device Coupling and Decoupling

01 - Coupling TV and Tablet

Created by Manuel Eisele, last modified by Christoph Ortner on 12.03.2014

User Story

Coupling TV and Tablet					
No		Priority	1	Size	tdb
HbbTV	X	Tablet	X	Web	
Story Narrative					
As a	user [consumer] with an activated GetTVivid account				
I want	to couple an HbbTV device with a tablet computer by photographing a QR-code from the TV or by typing in a displayed TAN				
So that	I can use the tablet as a second screen device for displaying information and navigation				
Acceptance Criterion 1					
Given	that the mobile device is already known to the system as coupled device				
When	the user [consumer] submits a valid code				
Then	refuse the coupling and display a corresponding message or offer the two options: (1) cancel, (2) overwrite existing tablet-TV-combination				
Acceptance Criterion 2					
Given	that the coupling has been successful				
When	the user [consumer] submits a valid code				
Then	display a corresponding message or forward the user [consumer] directly to the starting page				

Message content

Code: m TAN or QR-code

Comments

The Coupling is performed once and is permanent. The QR code should pop up on the TV screen at the first log-in. The option to couple two devices should still be reachable via the Account Settings

02 - Decoupling TV and Tablet

Created by Manuel Eisele, last modified by Bernhard Feldbacher on 12.03.2014

User Story

Decoupling TV and Tablet					
No		Priority	1	Size	tbd
HbbTV		Tablet		Web	X
Story Narrative					
As a	user [consumer] with two coupled devices				
I want	to decouple my devices				
So that	I can couple them with other devices.				
Acceptance Criterion 1					
Given	there is an existing coupling between the users [consumer] TV and tablet				
When	the user [consumer] logs into the web interface				
Then	the user [consumer] can decouple his devices.				

05 Offer-driven approach

01 - Posting a General Offer - Professional

Created by Manuel Eisele, last modified by Bernhard Feldbacher on 27.02.2014

User Story

Posting a General Offer - Professional					
No		Priority	1	Size	td
HbbTV		Tablet		Web	X
Story Narrative					
As a	user [provider] with an activated account to which I am currently logged-in to				
I want	to offer my services to other users [consumer] by adding them to the catalog via the web-interface				
So that	other users [consumer] can request them in an easy way at anytime				
Acceptance Criterion 1					
Given	that the user [provider] has completed the form correctly				
When	submitting the add-service request to the server				
Then	the service should be available to other users [consumer] immediately				
Acceptance Criterion 2					
Given	that the user [platform administrator] has defined, that all professional offer requests have to be approved by the user [platform administrator] or the user [platform administrator] has defined, that only offers by providers marked as trustworthy may appear in the catalog and the user [provider] has not been marked as trustworthy yet.				
When	a user [provider] submits the add-service request to the server				
Then	acceptance criterion 1 should be paused until the user [platform administrator] has accepted the request or the user [platform administrator] has marked the user [provider] as trustworthy.				
Acceptance Criterion 3					
Given	that the user [provider] has not assigned at least one performer for the service				
When	submitting the add-service request to the server				
Then	a message should be displayed and the "add" operation should be refused until at least one performer is assigned				

Message Content

Name:	m	Short name for the service offer, e.g. "room cleaning"
Description:	m	Free text for a more detailed description of the offered service
Location:	m	At the providers address, the consumers address or a defined address
Price:	o	Pricing information for the service
Category:	m	One element of the catalog tree structure to which the service should be classified to
Tags:	o	Tags which can be used for a free-text search
Timeframe:	o	General Timeframe when the service is available, e.g. daily from 8:00 AM to 4:00 PM
Performer:	m	List of possible performers for the service; at least one
Fields of expertise	m	What expertises does the company have; e.g. painter, decorator

02 - Responding to a General Offer – Professional

Created by Manuel Eisele, last modified by Bernhard Feldbacher on 18.03.2014

User Story

Responding to a General Offer - Professional					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	
Story Narrative					
As a	user [consumer]				
I want	to respond to a general offer				
So that	I get a professional service				
Acceptance Criterion 1					
Given	a user [consumer] and a user [provider] have a means to negotiate				
When	an agreement is reached				
Then	a legal contract is established.				

03 - Posting a General Offer – Informal

Created by Manuel Eisele, last modified by Bernhard Feldbacher on 02.07.2014

User Story

Posting a General Offer - Informal					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	
Story Narrative					
As a	user [consumer] with an activated account to which I am currently logged-in to				
I want	to offer my (informal) help to other users [consumer] by adding an element to the catalogue				
So that	other users [consumer] can request them in an easy way at anytime				
Acceptance Criterion 1					
Given	that the user [consumer] has not defined a receiver group or list				
When	submitting the offer to the platform				
Then	the offer should be listed in the catalogue structure for all users and be requestable for all users				
Acceptance Criterion 2					
Given	that the user [consumer] has defined a receiver group or list				
When	submitting the offer to the platform				
Then	the offer should be listed in the catalogue structure solely for the affected users [and they should be notified about the additional offer?]				

Message Content

- Name: m Short name for the service offer, e.g. "room cleaning"
- Description: m Free text for a more detailed description of the offered service
- Number of requests: o Limit the number of possible requests.
- Location: m At the providers address vs. at the consumers address
- Category: m One element of the catalogue tree structure to which the service should be classified to
- Tags: o Tags for free-text search.
- Timeframe: o General Timeframe when the service is available or not available, e.g. daily from 8:00 AM to 4:00 PM
- Recipients: o Group or list of consumers that are allowed to see resp. request the offer
- Local Criteria: o in my proximity (i.e. close perimeter of 500m)

04 - Responding to a General Offer – Informal

Created by Manuel Eisele, last modified by Bernhard Feldbacher on 18.03.2014

User Story

Responding to a General Offer - Informal					
No		Priority	1	Size	td
HbbTV	X	Tablet	X	Web	
Story Narrative					
As a	user [consumer]				
I want	to respond to a General Offer (Informal)				
So that	I can obtain the offer.				
Acceptance Criterion 1					
Given	a user [consumer] and a user [consumer] have a means to negotiate				
When	an agreement is reached				
Then	both sides are happy. 😊				

05 - Posting a Short-Term Offer

Created by Manuel Eisele, last modified by Christoph Ortner on 12.03.2014

User Story

Posting a Short-Term Offer					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	
Story Narrative					
As a	user [consumer] with an activated account to which I am currently logged-in to				
I want	to offer short-dated services and information to a defined or undefined group of other users [consumer] by sending a message to the platform				
So that	the other users [consumer] can be notified on condition that they are available to the system during the period of my message-availability				
Acceptance Criterion 1					
Given	that the user [consumer] has not defined a receiver group or list				
When	submitting the message to the platform				
Then	a predefined group should be used in order to avoid inconvenient broadcasting (e.g. user with the same address)				
Acceptance Criterion 2					
Given	that the availability time stamp is after the date time stamp				
When	submitting the message to the platform				
Then	a message should be shown and the operation refused until availability <= date				

Message Content

Name:	m	Short description in terms of a headline or a subject
Description:	m	Free text input option for a more detailed description of what is actually offered
Number of requests:	o	Limit the number of possible requests.
Date:	m	Timestamp defining when the offer will be executed
Availability:	m	Timestamp, defining how long the message will be available for spreading resp. be open for reactions of other users
Recipients:	o	Group or list of consumers that are allowed to participate defined by a list of criteria which is connected to the user [consumer]

Comments

Hide requests after they expired.

For further comments see [03 - Posting a General Offer – Informal](#).

06 - Responding to a Short-Term Offer

Created by Manuel Eisele, last modified by Bernhard Feldbacher on 18.03.2014

User Story

Responding to a Short-Term Offer					
No		Priority	1	Size	td
HbbTV	X	Tablet	X	Web	
Story Narrative					
As a	user [consumer]				
I want	to respond to a short-term offer				
So that	I can obtain the offer.				
Acceptance Criterion 1					
Given	that a user [consumer] requests the offer				
When	the user [consumer] agrees				
Then	both sides are happy. 😊				

07 - Posting a Joint Activity Offer

Created by Manuel Eisele, last modified by Bernhard Feldbacher on 18.03.2014

User Story

Posting Joint Activity Offer					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	X
Story Narrative					
As a	user [provider, consumer] with an activated account to which I am currently logged-in to				
I want	to offer joint activities in terms of multi-consumer and / or multi-variant appointments				
So that	other users [consumer] can vote / subscribe for 0..n suggested appointments				
Acceptance Criterion 1					
Given	that no availability limit is set				
When	submitting the joint activity offer to the system				
Then	the respective starting date of the offer is used as limit				

Message Content

Name:	m	Short description in terms of a headline or a subject
Description:	o	Free text input option for a more detailed description of what is actually offered
Category:	m	One element of the catalogue tree structure to which the service should be classified to
Tags:	o	Tags for free-text search.
Dates:	m	List of available dates
Prices:	o	List of prices, related to the different dates
Participants:	m	Number of minimum and maximum participants for each date
Scope:	o	Number of dates I am actually willing to offer (e.g. 5 suggestions but only 2 will actually take place at the end)
Availability:	o	Timestamp, defining how long the offer will be bookable for users (per date if limitations allow 1+ appointments per user)
Limitation:	o	Number of alternatives a user is allowed to vote for

08 - Responding to a Joint Activity Offer

Created by Manuel Eisele, last modified by Bernhard Feldbacher on 18.03.2014

User Story

Responding to a Joint Activity Offer					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	
Story Narrative					
As a	user [consumer] with an activated account to which I am currently logged-in to				
I want	to respond to a joint activity offer				
So that	I can obtain the offer				
Acceptance Criterion 1					
Given	that a user [consumer] requests the offer				
When	the user [consumer] agrees				
Then	both sides are happy. 😊				

09 - Modification of Offers

Created by Christoph Ortner, last modified by Bernhard Feldbacher on 17.03.2014

User Story

Modification of Offers					
No		Priority	1	Size	tdb
HbbTV	X	Tablet	X	Web	X
Story Narrative					
As a	user [consumer] with an activated account to which I am currently logged-in to				
I want	to update or delete my offer				
So that	it meets the actual necessities.				
Acceptance Criterion 1					
Given	that a user selects one of his existing offers				
When	he wants to update or delete this offer				
Then	the offer is modified accordingly.				

10 - System Generated (Joint) Activity Offers

Created by Christoph Ortner, last modified by Bernhard Feilbacher on 02.07.2014

User Story

Posting Joint Activity Offer				
No	Priority	1	Size	tbd
HbbTV	Tablet		Web	X
Story Narrative				
As a	user [administrator]			
I want	to generate Joint Activity Offers in regions with low or no activity			
So that	the activity in the system is augmented.			
Acceptance Criterion 1				
Given				
When				
Then				

Message Content

- Name: m Short description in terms of a headline or a subject
- Description: o Free text input option for a more detailed description of what is actually offered (e.g. How about going to the museum? Let's meet at 2 pm at the main entrance.)
- Category: m One element of the catalogue tree structure to which the service should be classified to
- Tags: o Tags for free-text search.
- Dates: m List of available dates
- Prices: o List of prices, related to the different dates
- Participants: m Number of minimum and maximum participants for each date
- Scope: o Number of dates I am actually willing to offer (e.g. 5 suggestions but only 2 will actually take place at the end)
- Availability: o Timestamp, defining how long the offer will be bookable for users (per date if limitations allow 1+ appointments per user)
- Limitation: o Number of alternatives a user is allowed to vote for
- Recipients: m List of users [consumer] that receive the system generated activity offer.

Comments

Based on Activity Profiling, Administrator input, proximity, weather etc.

PLUS: The system should be able to auto-generate joint activity offers based on history or templates (?). These auto-generated offers may be used by the platform administrator to motivate the users.
 ISOIN: Successful last offers created by user? How was the feedback to the last auto generated offer? Hard work, but it's possible

11 - Modification of an Agreement

Created by Bernhard Feldbacher on 18.03.2014

User Story

Modification of an Agreement					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	X
Story Narrative					
As a	user [consumer, provider] with an activated account to which I am currently logged-in to				
I want	to renegotiate an existing agreement				
So that	it meets the actual necessities.				
Acceptance Criterion 1					
Given	that a user selects one of his existing agreements for renegotiation				
When	he wants to modify the agreement				
Then	the appointment coordination starts again.				

06 Demand-driven approach

01 - Posting a General Demand

Created by Manuel Eisele, last modified by Bernhard Feldbacher on 02.07.2014

User Story

Posting a General Demand					
No		Priority	1	Size	td
HbbTV	X	Tablet	X	Web	X
Story Narrative					
As a	user [consumer] with an existing account to which I am currently logged-in to				
I want	to post a certain demand in terms of a service that I am generally looking for				
So that	other users [consumer, provider] can respond to my request by offering their help				
Acceptance Criterion 1					
Given	that the user [consumer] has defined his / her request properly				
When	when submitting it to the server				
Then	other users [consumer, provider] recognizing the demand should be able to answer directly				

Message content

Name:	m	Short name for the demand, e.g. "room cleaning"
Description:	m	Free text for a more detailed description of the offered service
Location:	m	At the providers address, the consumers address or a defined address
Category:	m	One or more elements of the catalog tree structure to which the service should be classified to
Tags:	o	Tags which can be used for a free-text search
User:	m	Groups or list of users [consumer, provider] who may receive the demand

Comments

See [03 - Posting a General Offer – Informal](#).

ISOIN suggests: Provide templates for demands that are based on a users service consumption history or the selected category. Important feature.
 PLUS: Match Making: Discuss options to automate the system to perform match-making between demands and offers (e.g. latest offers, categories).
 ISOIN: Again useful and possible.

01.1 - Posting General Demand (Alternative)

Created by Manuel Eisele, last modified by Christoph Ortner just a moment ago

In our opinion, the demand driven scenario needs an intermediary in order to eventually become really helpful for the consumer.

Therefore we've thought about a really simple approach how to make this functionality available.

Please see the PDF file for the first workflow idea. ([Demand Driven Approach 20140704 03 mei.pdf](#))

02 - Responding to a General Demand

Created by Christoph Ortner, last modified by Bernhard Feldbacher on 16.07.2014

Comments

See [02 - Responding to a General Offer – Professional](#) and [04 - Responding to a General Offer – Informal](#).

Escalation: If a demand stays unanswered for a certain period of time, the user's relatives or caretakers should be informed (e.g. via SMS or Email).

03 - Posting a Short-Term Demand

Created by Manuel Eisele, last modified by Christoph Ortner just a moment ago

User Story

Posting a Short-Term Demand					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	X
Story Narrative					
As a	user [consumer] with an existing account to which I am currently logged-in to				
I want	to post a certain demand in terms of a service that I am looking for				
So that	other users [consumer, provider] who have marked during the configuration that they are generally willing to help in such cases are notified and can respond to my request by offering their help				
Acceptance Criterion 1					
Given	that the user [consumer] has defined his / her request properly				
When	when submitting it to the server				
Then	other users [consumer, provider] recognizing the demand should be able to answer directly				

Message content

- : Category: m Category of the Short-Term Demand
- User: o Group or list of users, that may receive the short-term demand request

04 - Responding to a Short-Term Demand

Created by Christoph Ortner on 12.03.2014

See [06 - Responding to a Short-Term Offer](#).

05 - Modification of Demands

Created by Bernhard Feldbacher, last modified on 16.07.2014

User Story

Modification of Demands					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	X
Story Narrative					
As a	user [consumer] with an activated account to which I am currently logged-in to				
I want	to update or delete my demand				
So that	it meets the actual necessities.				
Acceptance Criterion 1					
Given	that a user selects one of his existing demand				
When	he wants to update or delete this demand				
Then	the demand is modified accordingly.				

07 User Accounts

01 - Account Status

Created by Christoph Ortner on 12.03.2014

User Story

Account Status					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	X
Story Narrative					
As a	user [consumer] of the GeTVivid system				
I want	to set the status of my account inactive or active				
So that	I don't receive any more notifications.				
Acceptance Criterion 1					
Given					
When					
Then					

02 - Configure Account for Demand Request Distribution

Created by Manuel Eisele, last modified by Christoph Ortner on 12.03.2014

User Story

Configure Account for Demand Request Distribution					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	X
Story Narrative					
As a	user [consumer, provider] with an activated account to which I am currently logged-in to				
I want	to mark a selection of categories [provider] or a selection of category-user group tuples [consumer]				
So that	demand-driven requests can be forwarded to me or notifications can be displayed in my account				
Acceptance Criterion 1					
Given	that a user [consumer, provider] wants to select a category				
When	he is not marked as trustworthy by the user [platform administrator]				
Then	requests shall not be forwarded				

Message Content

- Categories:** m Categories the user is generally interested into resp. is willing to help
- User:** [o] Group or list of users from whom I am willing to receive notifications resp. requests
- Availability:** o Timeframes during which a user is generally available or not available

03 - Role Achievements

Created by Christoph Ortner, last modified by Bernhard Feldbacher on 02.07.2014

Comments

Profiling, gamification, user roles (e.g. "the mechanic", "the transporter").

PLUS: Based on the users offer history achievements should be given to the user. E.g. someone who provided a service in category "repair" for three times should receive the achievement "repair-pal".
ISOIN: Some kind of personal tag to the user, not reduced to one. Own table related with the user. Which part of the system will populate the table?

08 Message Service

01 - One-To-Many (Mailinglist)

Created by Christoph Ortner, last modified by Bernhard Feldbacher on 16.07.2014

User Story

One-To-Many (Mailinglist)					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	
Story Narrative					
As a	user [consumer] with an existing account to which I am currently logged-in to				
I want	to send a message to a group of users [consumer] in my contact list				
So that	those are informed.				
Acceptance Criterion 1					
Given	the user [consumer] is logged into the system				
When	he or she selects users				
Then	the message shall be sent to the selected users [consumer].				

Message Content

- : User Identifications: m A list of identifications for already registered and activated accounts
- Message: m A message to the selected users.

Comments

The recipients should not know about the other users involved and should not receive their answers.

Definition **group**: Groups can be auto-generated by the system (e.g. "Nearby") or defined by the user (e.g. "Best Friends).

Contact list: Like Skype, with authorization from both sides.

How can we avoid that the users only chat, but don't use the demand/offer system.

Also have a look at [Minutes Bilateral Telco, 16.07.2014, \(EVISION, IRT, PLUS/ICTS, PLUS/COSY\)](#)

02 - Automatic Notifications

Created by Christoph Ortner, last modified by Bernhard Feldbacher on 02.07.2014

User Story

Automatic Notifications					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	
Story Narrative					
As a	user [consumer] with an existing account to which I am currently logged-in to				
I want	to receive automatic notifications				
So that	I am able to keep track of interesting topics, my medication, the weather etc.				
Acceptance Criterion 1					
Given					
When					
Then					

Comments

- Selection
- Notification

E.g. Weather forecast, news according to a selected list of categories, location etc.

ISOIN: Selection of news feeds, weather forecast: More a preference setting where you can select which kind of news you want.

E.g. There is no activity on a certain account for some days. The user [administrator] shall receive a notification (via profiling)?

Should the user [consumer] report any longer time away from the system (vacation, hospitalization etc.) to the system or the user [administrator]?

E.g. Reminders: Confirm reminders e.g. through pressing a key on the remote.

03 - New Offer/Demand Bubble

Created by Bernhard Feldbacher on 16.07.2014

User Story

New Offer/Demand Bubble					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	
Story Narrative					
As a	user [consumer] with an existing account to which I am currently logged-in to				
I want	to have an indicator on the start page				
So that	I am informed if there are any new demands or offers.				
Acceptance Criterion 1					
Given	a new offer or demand was created by any user				
When	I look at the start page of the application				
Then	an indicator should pop up (e.g. a red bubble with the number of new offers/demands in it) without reloading the page.				

Comments

: The user should be able to see, that the system is alive.

04 - Contact and Group Management

Created by Bernhard Feldbacher on 23.09.2014

User Story

Contact and Group Management					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	
Story Narrative					
As a	user [consumer] with an existing account to which I am currently logged-in to				
I want	to add users [consumer] to my contact list and organize them in groups				
So that	I can quickly select recipients for messages or offers.				
Acceptance Criterion 1					
Given	the user [consumer] is logged into the system				
When	he or she selects a group in the contact management and searches for a user [consumer]				
Then	the user [consumer] is added to the selected group in his or her contact list.				

Message Content

User: m The user to add to the current user's contact list.
 Group: m The group in the current user's contact list the selected user is added to.

Comments

Nuremberg:

Definition of a **group**: Groups can be auto-generated by the system (All Users, My Contacts, Family, Friends) or defined by the user (e.g. "Best Friends").

Contact list: There is no authorization required to see a user's contact details. All users are displayed with their full names in contact lists.

Also have a look at [Minutes Bilateral Telco, 16.07.2014, \(EVISION, IRT, PLUS/ICTS, PLUS/COSY\)](#)

09 Calendar

01 - Create a Calendar Entry

Created by Christoph Ortner, last modified by Bernhard Feldbacher on 18.03.2014

User Story

Create a Calendar Entry					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	X
Story Narrative					
As a	user [consumer] with an existing account to which I am currently logged-in to				
I want	to create a new appointment or to-do in my personal calendar				
So that	it helps me to organize my agenda.				
Acceptance Criterion 1					
Given					
When					
Then					

Message content

Title	m	Title of the appointment or reminder
Description	o	Short description
Type	m	Appointment or Todo (e.g. take a pill)
Reminder	o	An alarm that displays a notification on the clients.
Start time	m	Date-Time when appointment starts or todo has to be done.
End time	[o]	Date-Time when appointment ends. Only for appointments.

Comments

Reminders: Confirm reminders e.g. through pressing a key on the remote. See [02 - Automatic Notifications](#).

02 - Modify a Calendar Entry

Created by Christoph Ortner, last modified just a moment ago

User Story

Modify a Calendar Entry					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	X
Story Narrative					
As a	user [consumer] with an existing account to which I am currently logged-in to				
I want	to modify or delete an existing appointment or to-do in my personal calendar				
So that	it meets the actual necessities.				
Acceptance Criterion 1					
Given	the calendar entry was created by the user [consumer]				
When	he or she tries to modify it				
Then	modification shall be executed.				

Message content

Title	m	Title of the appointment or reminder
Description	o	Short description
Type	m	Appointment or Todo (e.g. take a pill)
Reminder	o	An alarm that displays a notification on the clients.
Start time	m	Date-Time when appointment starts or todo has to be done.
End time	[o]	Date-Time when appointment ends. Only for appointments.

Comments

Discuss possibility to shift agreed appointments through negotiation. [?](#)

If the calendar entry was created automatically via an offer, then the offer itself has to be renegotiated.

03 - Confirm a Reminder

Created by Bernhard Feldbacher, last modified on 02.07.2014

User Story

Confirm a Reminder					
No		Priority	1	Size	tbd
HbbTV	X	Tablet	X	Web	
Story Narrative					
As a	user [consumer] with an existing account to which I am currently logged-in to				
I want	to be reminded at the specified time				
So that	don't forget an appointment or to-do.				
Acceptance Criterion 1					
Given	that the reminder is triggered by a to-do (take the pills)				
When	the to-do is finished				
Then	I have to confirm via answering the delayed (1 minute) displayed question.				

Message content

: Comments

PLUS: In the calendar suggested (joint) activity offers could show up.

ISOIN: Exactly like all other suggestions. No problem.

10 Interaction Modalities

Comments

Interaction between/on mobile client and HbbTV client.

PLUS: The contrast of the user interface could be adapted to the day time and the users impairments.

If the user presses multiple buttons at one time on the tablet, the user interface could become larger.

ISOIN: Can give suggestions to the system to adapt. Model needs the information, can trigger the change.

The rest is UI. Very difficult to implement. Involves front-end partners as well.

11 User Profiling

01 - Suggested services based on the navigation history

Created by Alex Garcia, last modified on 10.04.2014

User Story

Suggested services based on the navigation history					
No		Priority	1	Size	tbd
HbbTV		Tablet	X	Web	X
Story Narrative					
As a	user [consumer] of the GeTVivid system				
I want	to see suggestions based on my navigation history				
So that	I can select different services quickly				
Acceptance Criterion 1					
Given	that the user has been looking for one or more services				
When	the user [consumer] goes to the services search screen				
Then	the GeTVivid system will show suggested services based on the user[consumer]'s navigation history				

02 - Suggested services based on purchase history

Created by Alex García, last modified on 10.04.2014

User Story

Suggested services based on purchase history					
No		Priority	1	Size	tbd
HbbTV		Tablet	X	Web	X
Story Narrative					
As a	user [consumer] of the GeTVivid system				
I want	to see suggestions based on my purchase history				
So that	I can select different services quickly				
Acceptance Criterion 1					
Given	that the user has purchased one or more services				
When	the user [consumer] goes to the services search screen				
Then	the GeTVivid system will show suggested services based on the user [consumer]'s navigation history				

03 - Suggested new services based on other users who have been looking for the same service as me

Created by Alex García, last modified on 10.04.2014

User Story

Suggested Services based on other users					
No		Priority	1	Size	tbd
HbbTV		Tablet	X	Web	X
Story Narrative					
As a	user [consumer] of the GeTVivid system				
I want	to see suggestions based on the services which other users have been looking for				
So that	I can select different services quickly				
Acceptance Criterion 1					
Given	that users have been looking for one or more services				
When					
Then					
Acceptance Criterion 2					
Given	that the user has been looking for one or more services which have been looked for by other users				
When	the user [consumer] goes to the services search screen				
Then	the GeTVivid system will show suggested services based on the user [consumer]'s navigation history				

04 - Suggested new services based on other users who have purchased the same service as me

Created by Alex García, last modified on 10.04.2014

User Story

Suggested Services based on other users purchases					
No		Priority	1	Size	tbd
HbbTV		Tablet	X	Web	X
Story Narrative					
As a	user [consumer] of the GeTVivid system				
I want	to see suggestions based on the services which other users have hired				
So that	I can select different services quickly				
Acceptance Criterion 1					
Given	that users have purchases for one or more services				
When					
Then					
Acceptance Criterion 2					
Given	that the user has hired one or more services which have been hired by other users				
When	the user [consumer] goes to the services search screen				
Then	the GeTVivid system will show suggested services based on the user [consumer]'s navigation history				

05 - Suggest most purchased services

Created by Alex García, last modified on 10.04.2014

User Story

Suggest most purchased services					
No		Priority	1	Size	tbd
HbbTV		Tablet	X	Web	X
Story Narrative					
As a	user [consumer] of the GeTVivid system				
I want	to see most purchased services suggestions				
So that	I can select different services quickly				
Acceptance Criterion 1					
Given	users [consumer] have been purchasing one or more services which I purchased at least one of them				
When					
Then					

06 - Identity impersonation system

Created by Alex Garcia, last modified on 10.04.2014

User Story

Identity impersonation system					
No		Priority	1	Size	tbd
HbbTV		Tablet	X	Web	X
Story Narrative					
As a	user [consumer] of the GeTVivid system				
I want	to know if someone is impersonating my account				
So that	I can take security measures				
Acceptance Criterion 1					
Given	that one user has entered into the GeTVivid system				
When	the user has a non-standard behaviour				
Then	the GeTVivid system will notify this to the user account and the administrator of the platform				

07 - Terms abusive detector system

Created by Alex Garcia, last modified by Bernhard Feldbacher on 02.07.2014

User Story

Terms abusive detector system					
No		Priority	1	Size	tbd
HbbTV		Tablet	X	Web	X
Story Narrative					
As a	user [consumer] of the GetVivid system				
I want	to notify when I see an abusive terms service				
So that	the system will notify the abusive terms service to the user [consumer] and to the administrator of the platform				
Acceptance Criterion 1					
Given	that there are services in the GetVivid system				
When	there are abusive terms services				
Then	the GetVivid system will notify to the user [consumer] and to the administrator of the platform				

Comments

ISOIN: This one is related to all "Post Offer/Demand" user stories (e.g. check for bad words).

08 - Suggest services most popular in my area

Created by Alex Garcia, last modified by Christoph Ortner on 17.07.2014

User Story

Suggest services most popular in the user's[customer] area					
No		Priority	1	Size	tbd
HbbTV		Tablet	X	Web	X
Story Narrative					
As a	user [consumer] of the GeTVivid system				
I want	to know most popular services in my area				
So that	I can select a service quickly				
Acceptance Criterion 1					
Given	that there are services in the GeTVivid system				
When	these services are in the user's [customer] area				
Then	the GeTVivid system will suggest these services to the user [customer]				

Comments

PLUS: Is this limited to a certain category? Does not make much sense otherwise.

ISOIN: Depends on the call parameters.

ISOIN: Interface has to be discussed. What should be displayed where?