LLMCMSS

Long Lasting Memories

Central Management System

Functional and Implementation Specification

DRAFT 0.5

Author: Wolfgang Scherer

Date Issued: 27-JAN-2010

LLMCMSS.01 - LLM CMS Specification

This page and its child pages contain the specification work and results for the LLM Central management system (LLMCMS).

Contents:

Overview

The LLMCMS is part of the LLM system.

The LLM system, detailed in LLMTOS is the technical system providing the LLM service, also detailed in LLMTOS.

The LLMCMS shall provide all necessary over-all management functionality in the LLM system:

- Administering user account data (Password, user credentials)
- Administration of training data (programs, schedule, results, performance)
- Presentation of reports about training performance
- provide a local user interface for the LLM system user
- provide a remote user interface for local and remote system administration

LLMCMSS.01.00 Document Control

LLMCMSS.01 document properties	
Item	**Value**
document	LLMCMSS.01
Issue	0.5
History - 0.1	Created

History - 0.2	added LLMCMSS.08
History - 0.3	updated to Ref Arch V .03
History - 0.4	added LLMCMSS.09
History - 0.5	added document control and updated parts list

Nature of this Specification

The specification is intended to be 2 things in one:

1. A requirements specification, containing all the requirements for the LLM-CMS, functional, non-functional, etc.
2. An implementation specification, allowing to be handed over to an implementation software team to be the complete base to implement the software comprising the LLM-LUI, the CMS-LUI and the CMS components

By this approach, any traceability issues between requirements and implementation specification should be able to be minimized.

As a result, this approach adds further areas to the specification:

- requirements concerning specific structural elements of the implemented software. These may also result from project constraints and already existing components, e.g. the LUI already existing in the eHome system.
- Implementation and modularity hints and rules in every part of the specification

The specification builds on a Reference Architecture (open in new window for reference)

Functional Specification Areas

This specification specifies implementation requirements for 3 Components of the reference architecture architecture:

- The LLM-LUI
- The CMS-LUI
- The CMS itself including its RUI, accessing the opeerative components ILC, CTC, PTC and additional to-be-added components PLUS the LLM-Database to manipulate and investigate persistent data

Non-functional specification areas

The specification also specifies other areas:

- models and concepts for data management

- interfaces between components and (if necessary) internal to components
- volumetrics
- Performance
- Availability
- Safety
- Security
- Modularity
- Multi-Tenancy
- Use-Case: e-Home, Day-Care-Center, Clinical Environment, Home-Care Professional
- Scalability
- Accounting
- Evidence Collection & Archiving

Implementation and Architectural Areas

- LLMCMSS.05 - The LLM LUI Framework (LLF) specification

Optional System Features

This section lists optional system features and separates those included in version 1 of the LLM system implementation from those left for later versions.

Motivation for Optionality

This Specification describes part of the first version of the LLM system. The extent to which features can be implemented is determined by the project resources in the LLM project itself: project timeline and available development resources and funding.

Although the LLM project in principle tries to use existing components, there is a development effort necessary to develop the "glue" functionality. Especially the CMS and the LUI framework are part of this "Glue" functionality providing a more seamless look-and-feel and an extended amount of manageability of otherwise separated sub-systems. The LLMTOS describes a rather large number of ideas and concepts assisting in boosting user-friendlyness and lowering usage barriers, be it physiological or psychological.

However, as project timeframe is limited (the core aim of the project, the operational trial phase, has to commence as soon as possible) and so are person-hour-resources for software development and integration. All that said, it appears that not all usability features sketched in LLMTOS will make it into version 1 of the LLM system.

Optional parts not part of LLM System Version 1

For the current version of the LLM system the following parts are considered optional and, as of now, not part of version 1 implementation:

1. Avatar implementation
2. Gesture support for enhanced controllers like Wii-Remote or Multi-Touch
3. Voice Recognition

However, implementation of Version 1 shall not build up roadblocks or unnecessary difficulties for a later incorporation of these features in later versions of the system. Therefore, implementation must be with such expansion in mind. Designs shall consider hooks and other expansion mechanisms as far as the knowledge of possible integration of the later-to-be-added features is at the time of design. Reviews must contain checks for this expasion mechanisms while keeping overhead low in order not to waste resources where they should be saved.

Optional part included in LLM System Version 1

However, of all the optional parts there may be a few that are indespensable when it comes to at least providing the least required level of user-friendlyness and adaptation to the needs of the target clientéle. As of the time being, these are:

1. elimination of keyboard typed input for authentication: logging on by contactless ID card.
   However, there are methods existing for authentication by security devices like fingerprint-scanners or other devices. These methods shall be used along with the software support provided in the target operation system(s).
   To further ease acceptance, mobile device methods like NFC/GSM-interoperability shall be checked in conjunction with movement detection also feasable with these devices.

Conventions in this Specification

Where this specification is referenced, it will be called **LLMCMSS**.

As this work is a specification, it is extremely important to cater for traceability. By tracebility, we understand the connections between different requirements and between the different types of requirements. These connections allow to determine if a requirement is caused by another one or is a pre-requiisite for another one. In order to establish these connections, all parts of this specifications are uniquely numbered by a hierarchical naming system. These numbers are prefixed by the ID of the specification itself.

An example for such a requirements ID is: LLMCMSS.02.01.02 which refers to the 2nd paragraph in the first chapter of the LLM LUI specification, being part 02 of this specification (LLMCMSS).

Other referenced documents will be used in a similar, appropriate manner.

When it comes to references to an external document (external means not part of the LLMCMSS), only a specific issue of the respective document will be referenced. If it is impossible to rely on a consistent version numbering and administration in the external document, a snapshot of the external document will be taken and kept as a copy with LLMCMSS.

Sources and additional information for this Specification

[LLMTOS] D3.1 LLM technical and operational specifications Issue v1.9 at the files section of Work Package 3 of the LLM site. There are tracebility issues in connection with this

document as it may change in the future. To have at least one stable reference point, issue 1.9 will be tried to be marked up and linked with requirements items in the LLMCMSS. To assist this case, the document will be referenced here as **LLMTOS *(trace/link Work to be continued)***.

Derived Requirements for other Parts of the LLM System

At numerous places in this specification, the specified functionality requires pre-requisites or certain functionality from other parts of the LLM system. At the time of writing for this specification, it is not known if the referred other parts of the LLM system are able to fulfil these requirements. One exemption from this is the LLM-DB, because its functionality for the time being is well defined by the LLM web-service reference. Anyway, the requirements to other system parts will be marked with "ENVREQ:" to denote a requirement of the LLM CMS against its environment. At review of this specification the ENVREQs will have to be checked and appropriate work packages scheduled to implement tham if not yet existing.

Parts of this specification

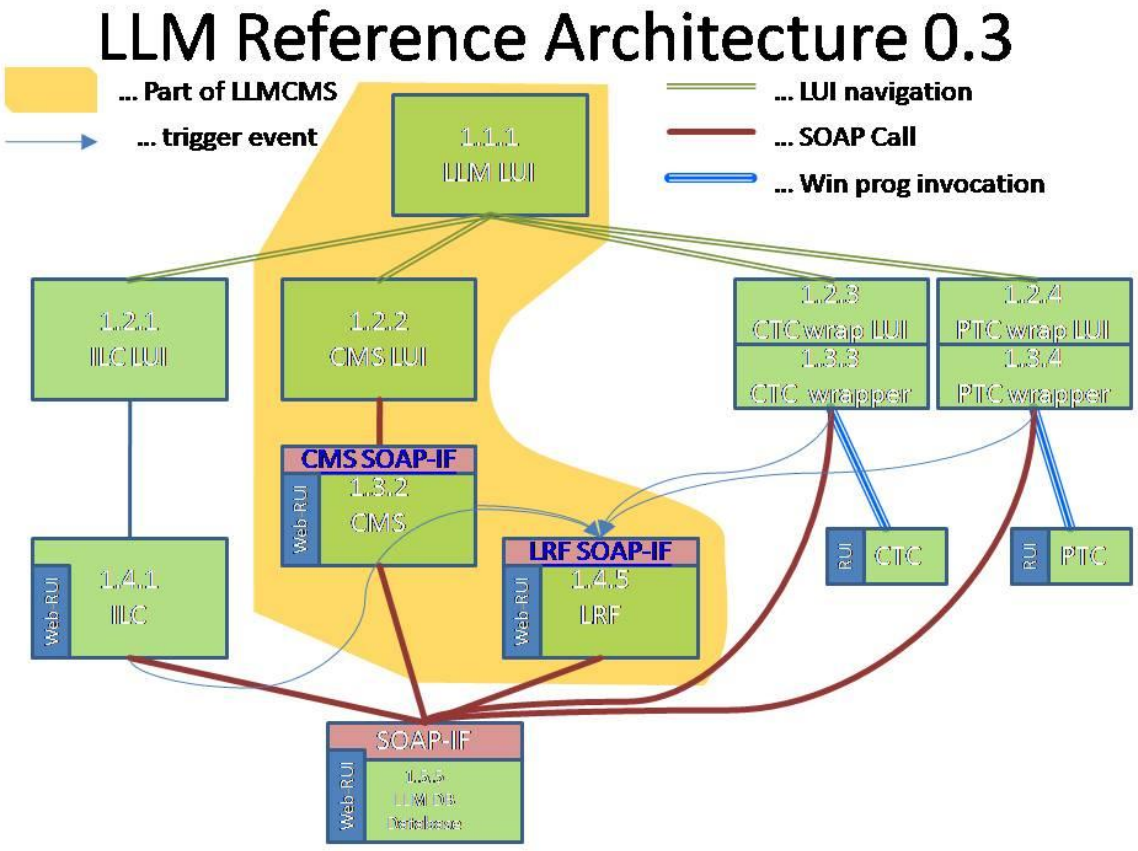
This specification consists of 10 parts. These parts, togethher with the documents referenced in the sources section above, comprise everything necessary for implementing LLMCMS. No other information source shall be necessary

1. LLMCMSS.01 - this part and top level of the hierarchy
2. LLM CMS Reference Architecture
3. LLMCMSS.02 - the LLM LUI specification
4. LLMCMSS.03 - the CMS LUI specification
5. LLMCMSS.04 - the CMS service specification
6. LLMCMSS05 - the LLF - LLM LUI Framework - specification
7. LLMCMSS.06 - LLM CMS Use Cases
8. LLMCMSS.07 - LLM CMS Aspects: Configuration, Security, Modularity, Performance, Volumetrics
9. LLMCMSS.08 - LLM reporting and user feedback data and flow models and implementation concepts
10. LLMCMSS.09 - CMS requirements in LLM-Database

LLM CMS reference architecture

In order to refer to functional entities in the CMS specification, a reference architecture of the LLM system is assumed. The LLM CMS specification will refer to parts of this reference architecture when it comes to specify attibutes or behaviour or interfaces between entities.

LLM CMS Ref Arch document properties	
Item	Value
Document	LLM CMS reference architecture
Issue	0.4
History - 0.1	Created with reference architecture V 0.1
History - 0.2	Updated reference architecture V 0.2
History - 0.3	updated reference architecture 0.3
History - 0.4	added document control



The elements in the yellow shaded area in the picture are objects of the LLM CMS specification. All other objects are used as LLM CMS has interfaces to them or their interfaces are used by LLM CMS too.

Concepts in the reference architecture

This reference architecture has been chosen to reflect the most probable functional component landscape the LLM system (the LLM system is assumed to be the technical system providing the LLM service) will be composed of.

Seperated and not separated components: LUI and RUI

There are components that look similar but have completely seperate behaviour and implementation. Examples of such component "pairs" are the LUI and RUI components.

Each component has a LUI - Local User interface - and a RUI - Remote User Interface. The LUI is the user interface for the local user and performs user-centric tasks. The RUI is the user interface for the remote access user and the administrator and performs user-centric tasks as well as administrative, management, configuration and troubleshooting tasks. To a certain extent (detailed with each component having LUI and RUI) the LUI is a (rather small and simple) subset of the RUI.

The RUI is an built-in functionality of each component and as of LLM system architecture V 1.0 an HTML-based web user interface. It provides all user interaction the respective component is capable of. As a result, the RUI is tightly bound the the component itself and uses only loose coupling to other RUIs and therefore assumed as being part of the component itself.

The LUI is more simply and user-centered. All LUIs of a LLM-system are much more tightly and seamlessly integrated to provide maximum usability and comfort. Therefore, the LUIs share a lot of common functionality and are more tightly interwoven than the RUIs. Moreover, the LUIs are in some places "wrappers" around the "original" user interface of a component. As a result, the LUI of a component makes sense to be separated from the component.

Integration of already existing components

Moreover, already existing components, as CTC, ILC and (to become) PTC, shall be able to take part in this architecture as-is.

However, ILC, CTC and PTC are here referred to as placeholders for ***any*** ILC, CTC or PTC that fulfils the behaviour required for taking part in a functional LLM system. How this can be accomplished is detailed below in the respective section(s) covering these components.

LUI - Local User Interface and LLF - Local user interface framework

There are a number of LUI - Local User Interface - Components in the LLM reference architecture. The LUI is a user interface intended for the local, not technically skilled and, most probably, physically challenged, user.

For the purpose of the LLM CMS the assumption will be made that the eHome LUI already has proven to be able to meet the requirements for a LLM LUI. Furthermore, it is already existing and can be easily adapted to the functions required in LLM LUI.

Therefore, the eHome LUI software component and platform shall be adopted for all LUI components in the LLM architecture. The framework extracted from the eHome software package shall be called LLF - LLM LUI framework - for the sake of brevity and identification.

The LLF is a windows application running on a specific set of touch-enabled net-top intel x86 hardware and operating system platforms. It is specified in the LLF specificationpart of this specification.

Components of the reference architecture

The reference architecture is divided up into components each fulfilling a specific functionality.

The LLM-LUI

The LLM -LUI is the central local user interface to one LLM-system. It is the central entry point to all components of an LLM instance. The LLM-LUI shall allow access to:

- ILC-LUI - The local user interface of the independent living component
- CTC-LUI - the user interface of the cognitive training component
- PTC-LUI - the user interface of the physical training component
- CMS-LUI - the administration user interface for the whole LLM system

The LLM-LUI shall also present the single point of authentication for a user. After login into the LLM-LUI, all user credentials shall be carried on to the connected subsystems and their LUIs. A user shall only need to login once and then be known to all subsystems with all their data and access rights.

The process of making one LUI visible, most probably implemented by bringing its window to the front of the display screen, is named switching.

The LLM-LUI shall switch between the respective user interfaces as seamless as possible (depending on the ways viable in integrating the user interfaces of already existing components).

Switching shall not only be initiated by the human user, but also by a subsystem needing attention shall be able to bring its LUI to the front and attention of the user.

The ILC-LUI

The ILC-LUI is the user interface of the independent living component. It is integrated into the LLM-LUI.

As of the time of releasing this specification, the only available ILC implementation used in LLM is eHome . It is assumed that the LLM-LUI will use the same base technical

implementation as the eHome-LUI. Therefore, integration shall be as tight as possible and the ILC-integration into LLM-LUI shall serve as the blueprint of an "Integrated Application" in terms of LUI-interaction.

The CTC-wrapper-LUI

The CTC-LUI is a wrapper component around the real user interface of the cognitive training component. It is assumed to be integrated into the LLM-LUI.

The PTC-LUI

The PTC-LUI is a wrapper component around the real user interface of the physical training component. It is assumed to be integrated into the LLM-LUI.

The CMS-LUI

The CMS-LUI is the user interface of LLM-CMS, the Central Management System. The CMS-LUI covers a subset of all possible administrative activity of the CMS, namely the part important for the local user, like viewing statistics and results, changing password or other user parameters. More detailed administrative and management activties are handled by the CMS-RUI built into the CMS.

The CMS

The CMS is a web-service providing the "business logic" of the management functionality. It offers an Web-Service-API for a number of functional groups:

- session management (for LUIs and other up-stream components
- user data administration
- general configuration management

There can be any number of CMS instances per LLM-DB instance but one CMS instance only has access to one LLM-DB instance.

The LLM-Database / LLM-DB

The LLM-DB is the central data storage component of one LLM-system. It holds all persistent data centrally for one LLM-system. All other components are required to refer to the LLM-DB to store and retrieve persistent data.

The LLM-DB exposes the LLM-Web-Service-Interface built on SOAP for all data manipulation and retrieval.

The LRF - LLM reporting and feedback functionality

The LRF generates feedback and reports for all types of users when triggered by another application.

Any application activity, when it has data to be brought to the attention of users, can trigger a report (or feedback, when it comes to informing the user about the performance of a training

session) to be generated. This triggering is performed by invoking a trigger method on the LRFs web API. The LRF then produces the appropriate representation of the data requested.

The data the LRF works on is taken from the LLM-DB. The result, any type of appropriate information, usually (but not limited to) human-readable and displayable on a LUI or RUI, is also deposited in the LLM-DB. Input and output of the reporting/feedback process are held in the same area of the LLM-DBs storage, in order to make the result of one report available as input for another report.

LLMCMSS.02 - LLM LUI Functional Specification

The LLM system Local User Interface - LLM LUI - is the central entry point for the local user access to the LLM system.

In other parts of this document it will be referenced as LLMLUI or LLMCMSS.02.

LLMCMSS.02.00 Document Control

LLMCMSS.02 document properties	
Item	Value
Document	LLMCMSS.02
Issue	0.2
History - 0.1	Created based on Reference Architecture V 0.1
History - 0.2	Updated revised section format

LLMCMSS.02.01 General

1. The LLMLUI shall be based on the LLF - LLM LUI Framework
2. Access to any function shall be controlled by the rights a user has by their credentials

LLMCMSS.02.02 Usage Scenarios

LLMCMSS.02.02.01 System Start / Shutown /Session housekeeping

1. When switching on the LLM local terminal, the LLMLUI login screen shall be presented
2. When switching off the LLM local terminal, the user's session shall be locked and cancelled (timeout logout) after a configurable amount of minutes. Additionally, switching off or loss of power of the local terminal shall be reported as an alarm condition.
3. Inactivity/Logout timeout shall be configurable as a system item overridable by a user item
4. Inactivity timeout shall be made variable depending on the active screen(s)/window(s)
5. An application shall be able to "re-arm" the inactivity timeout (in case a longer activity does not require user LUI activity)
6. Sessions shall be saveable and recall-able, also from other local terminals in the same LLM system.

7. Sessions shall have unique IDs identifying one sesion uniquely in activiity logs and for audit purposes

LLMCMSS.02.01.02 Logging in

1. Login shall grant user access to the system and authenticate them by various credentials:
    1. enter username and password onscreen
    2. provide ID by contactless smart card (smarcard reader required on local terminal)
    3. provide ID from Wii-Controller or similar non-standard device by opened library API
2. It shall be possible to log into one of more LLM system instances
3. Alternatively, a user having rights in more than one LLM system instance may be automatically logged into all of them
4. Login screen shall be able to be pre-populated when provided with data from ILC or a trainer/care-person
5. ILC or other application shall be able to initiate logout upon movement detection or patient change situations

LLMCMSS.02.02.03 User Administration

1. LLMLUI shall allow to create a user account, provided creating user has appropriate rights
    1. new user account shall be pre-filled from a template, template shall be selectable at creation time
2. LLMLUI shall allow user account deletion, provided user has appropriate rights
3. user accounts shall have arbitrary number of attributes, LLMLUI shall take account for this fact:
    1. UserCategory: Senior, User
    2. UserRole: Relative, Therapist, Administrator, Unknown
    3. All attributes assigned by LLMDB web service interface
    4. Additional Attributes as seen fit to fulfil additional features:
        1. Multi-Tenancy
        2. Sub-Grouping
        3. Privacy Attributes
        4. Judisdiction imposed properties
4. unique user ID shall be constant across database versions and shall allow
    1. moving a user across database instances
    2. exporting / importing single or multiple users
    3. merging and splitting of LLMDB instances during re-organization
    4. manipulation of users on database level without breaking validity on web service level

LLMCMSS.04.02.04 Invoking xTC and other subsystems, switching LUIs

1. LLMLUI and all subsystem LUIs shall have a set of controls to switch to another subsystem (for this part, LLMLUI is assumed to be just another subsystem LUI to be switched to, which will hopefully prove as the intuitives way also for the user).
2. If the subsystem to be switched to is not already started, it shall be started when switching to it, creating a look-and-feel of "always there" which may be more convenient for the user perception.
3. each subsystem shall have its own window. Usually, only one subsystem (the one called to be "in front") will be visible at a given time. However, on larger screens more than one subsystem LUI will be visible with possible overlap, but with a clear notion of one being "in

front". Thgis shall be implemented by means of LLF, LLF using the appropriate OS library functions available in all target OSs for the local terminal platform.

4. subsystem LUI processes shall be able to request to be put "in front " in case of important information to be displayed.
5. Apart from being "in front", LUI windows shall be able to request attention by drawing on their windows anyway (also if not "in front", screen display may find another way) and by generating audible output, which shall be possible also for not-"in front" LUI windows. Audible prompts may be a convenient way anyway for appointments or schedules without any need to put the generating window it front if the sound messages contains the appointment or schedule information (text-to-speech).

LLMCMSS.04.02.05 Scheduling and appointments

1. As a LUI subsystem of its own, a scheduling and appointment-LUI shall be invoked and displayed at each logon.
2. The scheduler-LUI shall show a day/month/week/list view with appointments marked.
3. If an appointment is due, the schedule-LUI shall transfer control to the LUI the appointment is for:
    1. CTC and PTC for a scheduled training session, passing training schedule information to the respective xTC to automatically start with exercises and instructions
    2. ILC with additional information for invoking phone or video conversations for trainer's or doctor's appointments or to start conversations with relatives on their birthdays
    3. simply displaying a message or playing an audible or video announcment for general reminders
4. Editing schedules and appointments shall be done with the CMS and CMSLUI.
5. Schedules shall be stored in the LLMDB associated to their "owner" user account
6. Schedules shall be able to trigger workflows of activities to be executed in sequence. Such chains shall be implemented as chains of scheduled acticvities.
7. To enable chains of activities, there shall be a number of special "trigger" conditions apart from a certain one-time or repeating interval:
    1. At logon. Most probably linked to a specific local terminal or group of terminals. Also shall be qualifyable by the logon method, i.e. using an ID card or one of a set of ID cards.
    2. After completion of another activity, with optional time delay

LLMCMSS.04.02.06 Remote Control and Remote Assistance

1. A support person shall be able to share user's local terminal view or at least a set of windows and their state.
2. The support person shall be able to assist the user and perform actions on the LUI on their behalf in parallel with a audio or video conversation.
3. This remote support scenario shall help in implementing use cases like trainer-assisted PTC and CTC usage.
4. In a usage scenrio of workflow and remote control the support-person's local terminal shall have a pop-up with the user's local terminal address and shall be able to connect to the local terminal by clicking on a control in the pop-up.
5. Seeing the user's local terminal window, the support person shall also be enabled to know the cause for support by seeing the appointment pop-up.
6. Alternatively, provisions shall be developed to have cross-terminal and cross-session workflows

LLMCMSS.03 - LLM CMS LUI functional specification

This part of LLMCMSS provides the functional specification for the local user interface (LUI) for the LLM system central management system (CMS).

LLMCMSS.03.00 Document Control

LLMCMSS.03 Document properties	
Item	**Value**
Document	LLMCMSS.03
Issue	0.2
History - 0.1	Created due to Reference Architecture V 0.1
History - 0.2	- Updated cross-references to LLF and LRF in Reference Architecture V 0.3 - Details on LUI functions

LLMCMSS.03.01 CMS LUI functions

1. The CMS LUI shall provide all functionality to be performed by the consumer of the LLM service, also referred to as the **senior** .
2. Change Password
    1. This optional shall only be available if the user is entitled to it by their rights
    2. Password change shall require the old password and twice the new password
    3. This option shall NOT be available if authentication is done by alternative method like smart-card or sound recognition

1. Display xTC and other (medical, etc) available performance data and feedback information, prepared by the LRF, detailed in LLMCMSS.08
    1. Display shall be initiated either by notification or deliberate user selection
    2. User selection shall include selecting only reports applicable for the user. Applicability shall be performed by matching user report selection properties against report properties. For LLM V 1.0 there shall be a collection of properties in the user profile that shall be exactly equal to the same subset of report properties.
    3. Notification selected report display shall provide a hyperlink (button) on the notification page invoking exactly one report for display
2. Display and Edit xTC and other schedules (repeating) and appointments (one time)
3. Access to the functions shall be controlled by the security credentials of the user's session
4. Apart from the UI-functions listed above the user may be presented with other display pages/windows initiated by workflows, schedules and notifications
5. After login, a special "Login-workflow" shall be triggered in the CMS service component that may provide additional notification-triggered activity-prompts

6. Before logout, a "logout-workflow" shall be triggered by a "logout" event in the CMS service component
7. Activity timeouts shall supervise the users activity or in-activity and shall trigger corresponding events in the CMS service component to generate notifications to the user or also to other users or sessions:
    1. timeout responding to notifications
    2. timeout performing no activity for a certain interval
    3. periodic timeout as long as LUI session exists to check for available information or schedules

LLMCMSS.03.02 CMS LUI Implementation requirements

1. CMS LUI shall be based on the LLF, detailed in LLMCMSS.05
2. CMS LUI shall use the CMS service component, detailed in LLMCMSS.04 to perform each activity
3. CMS LUI shall allow for multiple sessions by multiple users to multiple LLMDB instances on one local terminal
4. The LLF platform and environmental specifications also apply to the CMS LUI

LLMCMSS.04 - CMS service component functional specification

This part of LLMCMSS specifies the functionality and behaviour of the LLM CMS service software.

The current issue status is DRAFT V 0.5.

The CMS is the core of the central management of the LLM system. It has no user interface, but provides a web service API for all of its functionality. The LLM CMS LUI uses this API to provide a user frontend to the management functions.

The CMS component provides also a Web-based RUI (Remote User Interface) including, aside from the functions provided locally by the CMS LUI, all enhanced administrations activity necessary for LLM CMS configuration management.

LLMCMSS.04.00 Document Control

LLMCMSS.04 document properties

Item	Content
Document(Part)	LLMCMSS.04
Issue	0.5
History - 0.1	Created
History - 0.2	
History - 0.3	
History - 0.4	
History - 0.5	- create user accounts from templates in config - move LLM-DB interface to LLMCMSS.09 - precised user account management - precised session management - some details on credentials - some details on workflows/schedules/notifications/event processing

LLMCMSS.04.02 CMS functions

1. In general, each provided API functionality shall also be provided at the web-RUI.

2. Every API function shall be protected by access security and shall be logged including security credential information and terminal name
    1. User credentials shall be kept in the session and associated with a terminal.
    2. credentials shall be valid only with the terminal kept in the session. Therefore, each API call needs to have a terminal ID with it. API calls not from real terminals must provide valid virtual terminal IDs.
    3. Sessions may be transferred from one terminal to another by a specific "change terminal" API call. This API call shall not be available to all users, only to special ones.
3. There shall be a generic strcuture editor API to perform low-level viewing and changing of LLM-DB PROPERTIES structure:
    1. Session shall keep "edit point" in the PROPERTIES structure
    2. navigation commands/methods to go up/down/previous(n)/next(n)/list(n) in the PROPERTIES tree
    3. commands/methods to add/change/delete current item
    4. commands/methods to export/import from download/upload XML file
4. User logon
5. Session management
    1. Create Session, associate with terminal (local/remote) and user account
    2. Lock/Unlock Session
    3. Dump / Restore Session
    4. Terminate / Shutdown Session
    5. Manage Session variables/properties (create/query/delete/associate)
    6. change terminal
6. user account management
    1. A user account may have arbitrary properties in hierarchical manner associated with it.
    2. Create Account
        1. There shall be a template for each user account created
        2. The template shall have default property values copied into the newly created account
        3. If there is no template name given, a default template, the name of which is configured in the property "/sysconfig/sysdefaults/default_user_template_name", shall be used. The hardcoded default for this shall be that no template shall be used.
        4. If there is no template applicable, the empty template will be used, this means, no properties will be assigned to the user account.
    3. Lock/Unlock Account
    4. Export / Import Account
7. Preferences management
    1. Screen readability
    2. Audio levels
    3. Language Settings
    4. skinnig
    5. avatar hooks and avatar state
        1. preparation for integration of Audio TTS notification reading
        2. preparation for integration of alternative forms entry methods:
            1. provide user credentials / authorisation information via:
                1. smartcard
                2. sound recognition
            2. provide dialogue box (yes/no/...) responses via:
                1. sound recognition
8. Equipment Management
    1. PTC Equipement and Settings

    2. CTC Equipment and Settings
    3. export / import settings
9. Statistics data management
    1. query logging/statistics data
    2. filter
    3. export / import
    4. delete
    5. space management
10. schedule/appointment processing
    1. create scheduled activities/appointments with repeat/interval/calendar characteristics
    2. Execute the scheduled activity or generate notifications when schedules are due
    3. optional / for later versions: pre-announcements and repeat announcements if not confirmed
11. workflow processing
    1. CMS component shall be able to manage workflows conformant to WMFC standards, preferrably XPDL
       There are a number of open source XPDL processors available, shall be checked.
       Events and Agents and Activities shall be mapped to the respective LLM object types
12. notification management
    1. CMS shall keep information about the state of users and sessions
    2. any application shall be able to issue a notification request for a user or a specific session of a user via CMS SOAP API
       A notification shall be a displayable entity and shall have properties structuring the data and giving hints about presentation of the data
    3. CMS shall display the notification in the appropriate window and requested way plus bringing the window to the front if requested
    4. CMS shall keep track of confirmation of a notification (press a button when notification is read) if requested and perform follow-up activity in case of no confirmation from the user
    5. Notification shall be able also to take alternative forms additionally to local terminal display: send email, forward events to other applications and/or workflows
13. general functions for schedule/workflow/notification management
    1. General event API
       There shall be set of methods to inject events into the CMS subsystem.
       Events shall have arbitrary properties that can be used as mapping/routing criteria or to carry information up to displayable text or processing instructions
    2. Event mapping
       Events shall be able to be mapped to activities inside the CMS component configurably depending on property values
    3. Timing facility
       There shall be a function to execute activities based on absolute or relative time/date values or intervals. This facility shall be able to interact with the calendar/schedule management to perform activities and generate notifications based on schedule/calendar entries.

LLMCMSS.04.03 CMS function interfaces

The CMS function block has 4 interfaces:

1. Web-Service interface provided to be used by CMS LUI or other CMS function consumers
2. Web-Service client to LLM-DB

3. local Filesystem for local configuration, basic logging/auditing and data caching
4. web-application environment for basic bootstrap configuration

LLMCMSS.04.03.01 CMS function provided web service interface

The CMS function block shall expose a web service interface providing all the functions listed in LLMCMSS.04.01.

LLMCMSS.04.03.02 LLM-DB functions needed by CMS

All data structures and functions required by the whole CMS subsystem are kept together in LLMCMSS.09

LLMCMSS.04.04 CMS implementation issues

1. CMS shall be implemented as a web-application providing SOAP web services running in a JEE container
2. CMS shall be implemented stateless, only using internal structures for buffering, persistent storage shall be kept in LLMDB, most probably in the sessions context.
3. Instantiation and life cycle management
4. domain scope
    1. one instance of CMS shall be associated with one administrative domain, i.e. one instance of LLMDB. However, multiple database connections shall be possible to address redundancy issues

LLMCMSS.05 - LLM Local User Interface Framework (LLF) Specification

This part of the LLM CMS Specification specifies the framework components on which all local user interface components - LUI - shall be built upon.

LLMCMSS.05.00 Document Control

LLMCMSS.05 document properties	
Item	**Value**
Document	LLMCMSS.05
Issue	0.3
History - 0.1	Created
History - 0.2	Added Callback Web Service Interface
History - 0.3	added document control

LLMCMSS.05.01 LLF platform

1. The LLF shall operate on a local terminal platform
2. The local terminal shall have following attributes:
    1. Hardware platform(s):
        1. ASUS nettop as in eHome installations
        2. Desktop WinXP/Win7 PC specified for CTC software
    2. Operating System Environments: Windows variants:
        1. Win XP 32 bit
        2. Win 7 32 bit

LLMCMSS.05.02 LLF Functionality

1. Basic Operation Mode: The LLF shall present the user a dialogue with a collection of controls, displayed in a window. Controls may have varying graphical design and functionality.
2. Multi-Window: The LLF shall be aware of and being able to control multiple windows displaying user-I/O for multiple applications. These applications may be LLF-based or completely different, the only thing in common being the same local terminal (i.e. display/desktop)
3. Navigation Controls: There shall be controls to lead to another dialogue with a different set of controls. The new dialogue shall be able to be displayed:
    1. in the same window like the invoking dialogue, replacing the invoking dialogue
    2. in a new window. In this case, modality shall be avoided as it is generally too complicated to understand for a user. If the "calilng" dialogue has to wait for the called dialogue to be completed, the called dialogue shall be opened in the same window like the calling one, returning to the calling one after completing all activity in the called dialogue
4. Multi-Window Control Action: Activity by a control in one window shall be able to affect configuration and appearance in another window. In order to limit cohesion, impact shall be able to be limited to a specific control in the target window by configuration.
5. A control shall be able to perform arbitrary functionality, especially performing actions on instances of Web-Service-Interfaces. Possibilities may be limited by the fact that a control is executed in the context of a specific platform. The activity carried

out may be intended to affect a software component running on a different physical or logical platform or a seperated security realm. In all such cases, the LLM architecture calls for interaction via a web service interface.

6. There shall be a set of controls capable of "wrapping" already existing software applications running on the same platform. Depending of the ability of such applications, there may be different types of interactions and therefore different control implementations to interact with such an application:
    1. "Uncooperative Application": Such an application cannot be controlled in any way by the LLF control except for invoking it with an operating system call. The control shall at least monitor the existance of the invoked process and be able to modally block the LLF window during execution of the application or allow parallel execution. However, a set of safety and stability measures must be implemented:
        1. A timeout for modal or quasi-modal application windows to "escape" in case of a locked up application. Such application windows would prevent other activities while executing. If such an application does not terminate properly, all LUI would be locked up. Such a case needs to be avoided under all circumstances.
        2. process supervision for the application to detect creashed application processes
        3. measures to communicate the application status back to LLF to update the state of controls and other state-induced data
    2. "Cooperative Application": Such an application at least supports bringing another LLF window to the front of the display when the latterrequests it due to an alarm, notification or similar. This requires amending the application, but only to a certain extent.
    3. "Integrated application": such an application can be run inside a client window created by the LLF instance and therefore fully blended with the LLF look-and-feel. A model for such a mechanism would be the Internet Explorer engine embeddable into any .NET application.
7. Apart from the interaction on the GUI level, other integration needs to be taken care of:
    1. User credentials and authentication. The invoked application needs to run within the security allowances of the logged-in user on the LLF level.
    2. Data storage. The invoked application shall be enabled to place generated data in the LLM-DB via the WS-API
    3. Configuration. Like data storage, the invoked application needs to be enabled to take configuration information from the LLM-DB via the WS-API.
    4. The Data storage and Configuration Issues can be resolved in the worst case by "brute force wrappers": Such a wrapper would wrap around the application to be invoked and, at the start, retrieve the configuration and input data from the LLM-DB using session information and credentials. It would generate the input data in a format suitable for the invoked application (e.g. config files) and then invoke tha application. After completion of the application, the wrapper would collect all output (e.g. from statistics files etc.) and stuff it into the LLM-DB. This looks like a very crude process but could be the last resort with very non-cooperative application the LLM-project has no control over.
8. Call-Back Functionality: The LLF shall allow for an external application to control certain behaviour of the local terminal display:
    1. LLF shall offer a SOAP-API for each LLF instance on a local terminal
    2. API shall allow to create a window

3. control appearance and content of a certain control in a certain window
4. bring a specific window to the front

LLMLCMSS.05.03 - LLF Implementation and Modularity

1. LLF Implementation Code and Application Implementation Code shall be separately developed and maintainable. A system of dynamically loadable libraries (DLLs) along with a data-driven configuration data set shall be employed to achieve configuration flexibility, expandability and separation.
2. A sensible application of an MVC-(Model/View/Controller)-Pattern shall be followed. However, it is a definitive NON-GOAL of LLF to greatly re-design the basis software framework taken over from eHome.
3. Types of controls and other visual elements shall be able to be packaged separately and deployed independently. Multiple packages shall be able to be deployed into the same LLF instance and be distinguishable by a prefix/namespace-paradigm.
4. LLF shall allow for separated 3rd-party code libraries to be used by control implementations. One example for such a library would be a web-service client-library to access the LLM-DB or to access other LLM functional components
5. The variant of implementing LLF as a web-browser based frontend shall be kept in mind and re-evaluated at every new release of LLF. Existing CTC implementations already have web-frontends. Component-based web-application frameworks like Apache Wicket provide a large number of GUI controls providing native-like look-and feel by freeing LLF design from tasks and sotware components that are already existing in the market. Moreover, implementations like Apache Wicket are Open Source and free software.

LLMCMSS.05.04 - LLF Configuration Management

1. There shall be multiple LLF configurations per HW/OS-platform instance. These may be named "configuration" or "instance". One instance shall be configured by one set of configuration data and shall use one specific version of software components. Instances shall be able to be functionally and security-wise isolated from each other.

LLMCMSS.06 - LLM CMS Use Cases

This part of LLMCMSS specifies LLMCMS functionalities required by use cases as far they are concerning the LLMCMS.

The use cases are taken from LLMTOS.05 - LLM Operational Specifications.

LLMCMSS.06.00 - Document Control

LLMCMSS.06 document properties	
Item	**Value**
document	LLMCMSS.06
Issue	0.2
History - 0.1	Created topics based on LLMTOS
History - 0.2	added document control

LLMCMSS.06.01 - LLM Service Scenarios - LLMTOS.05.01

<to be completed>

LLMCMSS.06.02 - LLM Use Case Scenrios - LLMTOS.05.02

<to be completed>

LLMCMSS.06.03 - LLM Use Case Model - LLMTOS.05.03

<to be completed>

LLMCMSS.07 - LLM CMS Aspects: Configuration, Security, Modularity, Performance, Volumetrics

This section of LLMCMSS specifies all non-functional aspects of the LLMCMS subsystem under the following viewpoints:

1. configuration
2. security
3. modularity
4. performance
5. volumetrics

LLMCMSS.07.00 - Document Control

LLMCMSS.07 document properties	
Item	**Value**
document	LLMCMSS.07
Issue	0.2
History - 0.1	Created, list of aspects
History - 0.2	added document control

LLMCMSS.07.02 - Aspects Details

1. Configuration
2. security
3. modularity
4. performance
5. volumetrics

<to be completed>

LLMCMSS.08 - LRF functional specification

This is section 8 of the LLM CMS specification. It covers the area of how reporting and user feedback shall be handled and presented.

Reporting and Feedback are performed by the LRF (LLM Reporting and Feedback) component in the LLM system.

LLMCMSS.08.00 - Document Control

LLMCMSS.08 document properties	
Item	**Value**
document	LLMCMSS.08
Issue	0.2
History - 0.1	Created due to RefArch 0.3
History - 0.2	added document control

LLMCMSS.08.01 LRF Motivation and driving factors

This subsection does not contain requirements, all requirements that are derviced from the deliberations in this subsection are explicitely stated in the following subsections.

There are a number of factors influencing the way reporting shall be done and the software providing this information shall be structured:

1. Training components (PTC, CTC) deliver isolated, non-weighted and un-interpreted data points and samples collected during single or multiple training exercises and sessions.
2. There are different types of consumers for the information to be distilled from all this training data:
    1. The users (seniors) doing the training themselves
    2. The trainers and coaches
    3. The medical responsible personnel
    4. The relatives of the users
3. All consumers need different compilations of the information
4. Information needs to be weighted and compared against different benchmarks and thresholds:
    1. Absolute Values
    2. Relative to peer / reference group
    3. relative to own history
    4. in the context of medical status
    5. in environmental context
        1. weather
        2. living environment

        6. synchronized with other flows of events:
            1. medication
            2. personal events
            3. time of day/year/season
    5. The way this performance information has to be presented optimally also may vary due to several factors:
        1. type of senior personality
        2. psychological stability
    6. The nature and format of presentation and also the information extraction algorithms may vary also amongst installation sites and employed training equipment
    7. All above parameters will most probably also vary at least during the course of the trial phase
    8. There may be the need for testing the reception of various forms of performance feedback and collecting this type of "meta-feedback" information (feedback about the type of feedback)
    9. As a matter of system and software stability as well as a consequence of the above factors plus the fact that not all (if any at all!) types of feedback presentation and algorithms will be known at the first deployment of the LLM system, there needs to be a high flexibility and modularity in the împlementation and configuration of the preparation and presentation of training feedback information
    10. For Version 1.0 of the LLM system the following limitations in the feedback generations may apply:
        1. It is assumed that the training components already deliver weighted and interpreted results (good/fair/bad, etc.). So the LRF only has to do presentation, no decision-making.
        2. No correlation across the results of different training components and/or ILC. Also, no correlation with external data.

LLMCMSS.08.02 LRF Functional parameters

These requirements are partly derived from the factors found in LLMCMS.08.01, partly from LLMTOS, partly from other sections or external documents.

1. There shall be a reporting and feedback mechanism built as an autonomous component in the LLM system. This component shall be called LRF for brevity in the context of this specification
2. LRF shall work on measurement data deposited in the LLM-DB by training components and other sensors (Also ILC, as an example, shall be able to produce and deposit measurements data, for instance physical mobility detected by movement sensors)
3. There shall be 2 types of information calculated in the LRF:
    1. Reports:
       This shall be calculated values, statistics, charts, etc ...
    2. Feedback:
       This shall be personalized information for the target audience
4. Each generated information entity shall be qualified with the intended target audience and privacy, plus possibly additional qualification tags, plus security control.
5. Input data shall be taken from the LLM-DB
6. Generated information shall be stored in the LLM-DB, into the same structure as the inpout is taken from in order to generate feedback and/or reports based on provious

feedback/reports. Alternatively the LRF shall be configurable in the way from where to take the input data and to where to put the results.

7. Generation of information shall be able to be triggered by external events from other LLMSYS components and by internal schedules
8. Completion of information generation shall be able to generate events to trigger other activity like:
    1. user interface presentation and subsequent information processing. The generated trigger event shall be able to carry properties passed on from the event triggering the information generation. These properties shall include local terminal ID, session ID and all properties to bring the report data into foreground on the respective local terminal.
    2. Workflow event(s). For this purpose the triggering event shall be able to carry flow, step, event, session and user information necessary to identify the intended action in the workflow to be triggered by feedback generation.
9. There shall be "pluggable" statistics and report generation algorithms
10. Selection of applicable generation algorithm shall be able to be based on any data item available in the LLM-DB
11. There shall be multiple LRF-instances per LLM-System instance, each with their own configuration and loaded algorithm sets
12. Reporting results stored in LLM-DB shall be any combination of:
    1. Graphic Images: charts, traffic-light indicators, bar-graphs, ...
    2. single text or numerical values
    3. lists and tables
    4. general HTML
13. In order to leverage already existing developments and their evolution, existing processing libraries and components shall be employed, like:
    1. GNUplot, JFreeCharts, etc .. in the presentation level
    2. decision support frameworks and rules languages for the knowledge bases
14. All configurability shall be deposited also in the LLM-DB using configuration tables and properties

LLMCMSS.09 - CMS requirements in LLM-Database

This section summarizes all the data structures and access mechanisms used by the LLM-CMS in the LLM-Database.

All CMS subcomponents are clients for the LLM-DB components and store their persistent data there in structures described in the section.

LLMCMSS.09.00 - Document Control

LLMCMSS.09 Document properties	
Item	**Content**
Document (part)	LLMCMSS.09 - LLM-DB requirements by CMS subsystem components
Issue	0.2
History - 0.1	created as seperate section out of LLMCMSS.04.03.02, LLM-DB requirements.
History - 0.2	formatting, headers

LLMCMSS.09.03 LLM-DB interface required by CMS function

LLMCMSS.09.03.01 General PROPERTIES structure in LLM-DB

As some, if not most of the specific properties and attributes required by most of the components in the LLM-system CMS subsection are not yet known, expandability shall be built into the LLM-DB in order to keep the database structure stable and open for expansion.

1. There shall be a PROPERTIES structure in the LLM-DB with a hierarchical structure to be used for an enhanced version of the Name/Value pattern
2. properties shall be able to be grouped in structures like sections (as known from .ini files) or directories (analogy with file systems)
3. sections also should be properties having names, the value being the sub-structure, producing tree-like organisation
4. Each property entry item shall have a unique ID. Uniqueness shall apply inside one instance of LLM-DB. No uniqueness may be guaranteed when one item is deleted and another different one is created. In this case the unique ID may be re-assigned. Applications should

not rely on this uniqueness or, if required, take care of it on the application level. However, the unique ID shall be able to be used when traversing structures. If an entry has been deleted and is requested again by its unique ID, it is permissable to return a NULL value or to throw an exception (on the client API library side).
Additionally, unique IDs may change when exporting a sub-tree and re-importing it. This ensures the possibiity to import the subtree at another place in the higher hierarchy tree or importing it more than once, for instance in order to implement one type of templates.

5. A property name shall be a reference to a place in a certain property tree, and shall be able to be either of:
    1. absolute path in LLM-DB property tree, sections separated by "/" (preferable in the client library using path-separator variables)
    2. relative path to the "current" context the client is (using a context reference in the client library keeping track of the path context
6. A property value shall be one of:
    1. number - shall be able to hold 64-bit-FLOAT and 64-bit-INTEGER with auto-conversion if necessary
    2. string (unicode) - may be limited in size but no shorter than 1024 characters
    3. Document / Large String (CLOB)
    4. collection (=> strong link) of more properties
    5. reference (=> weak link) to one of: - recommended to be used only for certain cases as templates and shared properties
        1. collection in same DB-instance
        2. relative pathname to any type of item value
        3. absolute pathnameto any type of item value
7. properties references shall be usable in all other LLM-DB structure, however, LLM-DB may not need to be aware that some string is a property path
8. There shall be a client library in JAVA providing collection, properties and iterator interfaces to the properties structure
9. Methods in the client library shall keep a context to support buffering and maintaining a "current" place in the property tree
10. There should be methods in the client API to handle absolute and relative paths

LLMCMSS.09.03.02 Classes / Collections of items required and processed by CMS in LLM-DB

1. Seniors, enhanced by property handle for each entry
2. Users, enhanced by property handle for each entry
3. User rights, roles and permissions managed in property subtree named "auth". Master is account management in CMS components, all other clients must only read with robustness for changed structures between traversal reads.
4. Sessions, enhanced by property handle for each entry if already existing. If not yet existing, implemented as property subtree named "sessions". Sessions shall only be changed by the CMS component session management API, all other clients shall use this subtree as read-only with all the caution required by the fact, that one client may traverse the session tree for viewing purposes and the session management may delete the whole subtree or at least part of it.
5. Equipment Data: local terminals, Server Boxes, PTC devices, enhanced by property handle, if already existing and used by other components. If not yet existing, implemented as property subtree named "devices"
6. System Configuration Data, a property subtree named "sysconfig"
7. General Config-Items / Properties / Attributes in a property subtree named "config"
8. Activity-Information / Notification Data / Event Data in property tree(s)
9. Workflows in property tree(s) managed on a higher structural level inside the workflow processor, located in properties subtree named "workflows"

    1. Templates
    2. Instances
    3. Steps
    4. Agents, with relation to users
10. Schedules, Appointments (vs. ICalendar, ...), attached to users or devices or system general ("sysconfig") as properties subtrees named "calendar"