Managing Affective-learning THrough Intelligent atoms and Smart InteractionS

D3.1 The MaTHiSiS Smart Learning Atoms

Workpackage	WP3 – Smart Learning Atoms and Graph Tools
Editor(s):	Norbert BARICHARD, DXT Clément RODRIGUES-VIGUIER, DXT Thomas TECHENE, DXT Dorothea TSATSOU, CERTH Nicholas VRETOS, CERTH
Responsible Partner:	DXT
Quality Reviewers	Nicholas VRETOS, CERTH Mirela POPA, UM
Status-Version:	Final - version 1.0
Date:	Project Start Date: 1/1/2016; Duration: 36 months Deliverable Due Date: 31/12/2016 Submission Date: 13/01/2017
EC Distribution:	PU
Abstract:	This deliverable defines the concept of a Smart Learning Atom introduced in the MaTHiSiS project to describe an atomic and independent knowledge or skill to be acquired by a learner. Implementation details on the integration of this concept in the system are also provided through the description of the SLA library and it is Open API. In addition, the early front-end used to manipulate these entities is described to give a complete overview.
Keywords:	MaTHiSiS concept, skill, knowledge, competence, Smart Learning Atom, Open API, back-end, front-end

This document is issued within the frame and for the purpose of the *MATHISIS* project. This project has received funding from the European Union's Horizon 2020 Programme (H2020-ICT-2015) under Grant Agreement No. 687772

Related Deliverable(s)	*D2.2 - Full scenarios of all use cases (M9)* *D2.3 - Full system architecture (M6)* *D3.3 - The MaTHiSiS Learning Graphs (M12)* *D3.5 - Experience Engine (M12)* *D7.1 - Integration Strategy and planning (M6)* *D7.2 - MaTHiSiS platform, 1st release (M12)*

Document History

Version	Date	Change editors	Changes
0.1	05/11/2016	Norbert Barichard (DXT), Clément RODRIGUES-VIGUIER (DXT)	Initial version of the document. Front-end part.
0.2	11/11/2016	Clément RODRIGUES-VIGUIER (DXT)	Abstract, keywords, related deliverables, executive summary, introduction, conclusion
0.3	21/12/2016	Dorothea TSATSOU (CERTH),Nicholas VRETOS (CERTH)	Input sections 2 and 3 (back-end)
0.4	22/12/2016	Clément RODRIGUES-VIGUIER (DXT)	Finalization pass
0.5	26/12/2016	Thomas TECHENE (DXT), Norbert BARICHARD (DXT)	Integration of the feedback from peer-reviewers
0.6	28/12/2016	Dorothea TSATSOU (CERTH), Thomas TECHENE (DXT)	Alignment with the D3.3 and small updates
0.7	09/01/2017	Clément RODRIGUES-VIGUIER (DXT)	Formatting updates
1.0	13/01/2016	Carmen L. Padrón-Nápoles (ATOS) Clément RODRIGUES-VIGUIER (DXT)	FINAL VERSION TO BE SUBMITTED

The information and views set out in this document are those of the author(s) and do not necessarily reflect the official opinion of the European Union. Neither the European Union institutions and bodies nor any person acting on their behalf may be held responsible for the use which may be made of the information contained therein.

Table of Contents

List of Tables

List of Figures

List of Acronyms

Abbreviation / acronym	Description
ASC	Autism Spectrum Case
CGDLC	Career Guidance and Distance Learning Case
ITC	Industrial Training Case
LA	Learning Action
LAM	Learning Action Materialization
LCM	Learning Content Manager
LCS	Learning Content Space
LES	Learning Experience Supervisor
LG	Learning Graph
MEC	Mainstream Education Case
PA	Platform Agent
PMLDC	Profound and Multiple Learning Disabilities Case
SLA	Smart Learning Atom
US	User Space

Table 1: Definitions, Acronyms and Abbreviations

Project Description

The MaTHiSiS learning vision is to provide a novel advanced digital ecosystem for vocational training, and special needs and mainstream education for individuals with an intellectual disability (ID), autism and neuro-typical learners in school-based and adult education learning contexts. This ecosystem consists of an integrated platform, along with a set of re-usable learning components with capabilities for: i) adaptive learning, ii) automatic feedback, iii) automatic assessment of learners' progress and behavioural state, iv) affective learning, and v) game-based learning.

In addition to a learning ecosystem capable of responding to a learner's affective state, the MaTHiSiS project will introduce a novel approach to structuring the learning goals for each learner. Learning graphs act as a novel educational structural tool. The building materials of these graphs are drawn from a set of Smart Learning Atoms (SLAs) and a set of specific learning goals which will constitute the vertices of these graphs, while relations between SLAs and learning goals constitute the edges of the graphs. SLAs are atomic and complete pieces of knowledge [1] which can be learned and assessed in a single, short-term iteration, targeting certain problems. More than one SLA, working together on the same graph, will enable individuals to reach their learning and training goals. Learning goals and SLAs will be scoped in collaboration with learners themselves, teachers and trainers in formal and non-formal education contexts (general education, vocational training, lifelong training and specific skills learning).

MaTHiSiS is a 36 month long project co-funded by the European Commission Horizon 2020 Programme (H2020-ICT-2015), under Grant Agreement No. 687772.

Executive Summary

The current document presents one of the key concepts of MaTHiSiS, the Smart Learning Atom (SLA), and how it is integrated and edited within the MaTHiSiS ecosystem.

The concept of SLA was introduced in the MaTHiSiS project to allow a fine division of the knowledge, skills, or abilities to be transmitted in a learning scenario. This breakdown allows for a strong re-use of these learning atoms, and this approach is based on the fact that either in a school environment, as well as in a workplace, several new lessons are based on knowledge and skills that have already been acquired by the learner. This sets a common foundation on which all new courses and vocational trainings will be able to rely on.

After defining the concept of SLA, we provide the implementation details established in the MaTHiSiS system to represent and manipulate SLAs. The first pre-alpha prototype of the human-machine interface is also presented, which is the one that will be used during the first pilots phase. This interface will evolve over time depending on the feedback of end users and the progress of the development.

1. Introduction

This section provides detailed information about the purpose and structure of the document.

1.1 Purpose of the document

The current document is the deliverable ***D3.1 - The MaTHiSiS Smart Learning Atoms*** and describes one of the outcomes of work package ***WP3 - Smart Learning Atoms and Graph Tools*** and more specifically ***T3.1 - Educational Content development (Smart Learning Atoms)***.

It is a public document intended to describe one of the main concepts of MaTHiSiS: Smart Learning Atoms. It will be used as a reference during the development of the MaTHiSiS platform and as a documentation to explain how this concept is integrated into the system.

All partners, namely DXT, CERTH, NTU, and UM, have been involved in the definition of this concept and CERTH has done the implementation of the library and the Open API used to describe such entities. DXT has been working on the graph tools used to edit, compose, create, and re-use such educational content.

1.2 Structure of the document

This document contains the key sections detailed below:

- **Section** 1**: Introduction**
  This section clarifies the purpose and the structure of this deliverable and its context.

- **Section** 2**: Smart Learning Atoms**
  The second section is about the definition of the Smart Learning Atoms and why they have been introduced. Furthermore, an explanation will be exposed on how this concept is integrated in the MaTHiSiS system and which other concepts depend on it. Some concrete examples of Smart Learning Atoms are also given as a helper to understand the rationale behind them.

- **Section** 3**: Smart Learning Atom library implementation details**
  The third section describes more concretely how the concept has been integrated in MaTHiSiS. The data structure and the functionalities associated with Smart Learning Atoms will be described. The documentation of the Open API used to edit, update, and delete Smart Learning Atoms in the system is also given as a reference. Finally, the front-end parts where SLAs appear are described in order to complete the tour of the integration of this concept in all layers of the platform.

- **Section** 4**: Conclusion**
  This section presents the conclusions of the document and the next steps in the work related to ***T3.1 - Educational Content development (Smart Learning Atoms)***.

2. Smart Learning Atoms

This section details the logic behind MaTHiSiS core educational process innovation, namely the Smart Learning Atoms.

2.1 Objectives and definitions

Smart Learning Atoms (SLAs) are atomic and complete pieces of knowledge, competence and/or skills, which can be learned and assessed in a single, short-term learning process iteration from a learner. SLAs essentially comprise *primordial learning goals, constituents of more advanced learning goals, which cannot be further reduced to more primitive notions*. In a nutshell, they consist of the simplest of concepts pertaining to **what-to-learn** during an educational process.

SLAs comprise the basis of the novel learning approach of MaTHiSiS. Their standalone and self-contained nature allows learning scenarios designers to define re-usable, versatile learning entities, applicable to various learning subjects, which can collectively construct deeper learning goals. Smart Learning Atoms can be drawn from a library of pre-existing SLAs maintained by the MaTHiSiS platform, modified with custom attributes or engineered from scratch by the teacher/trainer. SLAs are the constructive elements of Learning Graphs (cf. Deliverable 3.3 *The MaTHiSiS Learning Graphs*), where already existing or new atoms are connected to composite learning goals, thus constructing a hierarchical network of the learning scenario objectives.

For instance, primordial skills/knowledge like "Object recognition", "Word recognition", "Spatial coordination" are primordial atomic learning goals, thus SLAs, that can be re-used in different learning scenarios and can constitute essential sub-components of many different composite learning goals. "Object recognition" and "Word recognition" for example might be needed in order to master "Vocabulary improvement", while "Object recognition" might also play an important role to another learning goal, such as "Subitizing".

Furthermore, breaking down learning goals to fundamental sub-components allows for non-linear and highly adaptive discretisation of the learning process, which does not have to follow a rigid collective and cascading style anymore. The hierarchical organisation of the learning objectives into atomic and composite units enables the learning experience to alternate its focus on mastering each atomic learning content constituent (i.e. the SLAs), which in term implies mastering the composite learning goals. Non-linear alternation between SLAs' training during the learning experience will ensure the adaptation of the learning process, based on the learner's particular apprehension abilities, learning style and uptake over the knowledge/skills to acquire, manifested as a competence score over the SLA.

This learner-centric scheme is also supported by the ability of the MaTHiSiS Platform Agents (PAs), acting as learning process facilitators, to deploy precise learning activities per SLA for each individual learner taking part in a learning scenario, in order to train the selected SLA(s) in each iteration of the learning process. To this end, each SLA is attached to one or more generic Learning Actions (LAs), which the MaTHiSiS system can materialise in different ways on the different PAs (cf. D3.5 *Experience Engine*).

2.2 Methodology and dependencies

In order to facilitate the self-contained nature of Smart Learning Atoms and at the same time their adaptability to different learner specifications, SLAs will take up two forms in the MaTHiSiS learning setting.

Unpersonalised, core SLAs will be maintained independently of Learning Graphs (LG) they might take part in, in order for them to be retrievable and re-usable in different learning scenarios (outlined by Learning Graphs). They will comprise of the universal properties of the particular competences/skills/knowledge pieces they entail, i.e. a name (identifier) and short description of what each SLA is about, while generic, PA-agnostic Learning Actions that can bring about the encapsulated competences/skills/knowledge will be attached to them. These structures reside on the MaTHiSiS Learning Content Space (LCS), detailed in Deliverable D2.3 *Full System Architecture*.

Personalised SLA instances will be created for each learner, upon the implementation of any learning scenario that involves these SLAs. These instances will incorporate a reference to their corresponding core unpersonalised counterparts and will bear a scalar weight per learner, indicating the uptake of the learner for this particular learning content item. Since SLA instances are again maintained independently from their connected learning scenarios, progress over them will affect any running or future scenario (Learning Graph) that the learner might be involved in. These structures reside on the MaTHiSiS User Space (US), detailed in Deliverable D2.3 *Full System Architecture*.

For technical reasons, the need to store a history of **runtime SLA instances** per learner has risen. While long-term personalised SLA instances will reflect the last state of each SLA instance per user, adaptation and personalisation needs to reflect upon and analyse the fluctuations of SLA weights across the course of a learner's experience. To this end, a historical record of runtime (denoted as 'rtm') instances of SLAs will be maintained, bearing a connection to the long-term (last state) personalised SLA instance for each learner, along with the reference to the particular session that an adaptation of the SLA instances' weight has occurred (or at least examined). Evidently, these structures also reside on the MaTHiSiS User Space (US).

It is easy to recognise that the SLAs' most prominent dependency is the Learning Graphs (LGs). Although SLAs can be maintained individually, they cannot be trained unless they take part in at least one concrete learning scenario (which is fulfilled by a LG). LGs are represented following the same logic as SLAs, therefore each SLA structure is related to their counterpart LG structures that they belong to, i.e. core SLAs are related to unpersonalised LGs, SLA instances are related to corresponding instances of these LGs for the particular learner and SLA runtime instances are related to their respective runtime LG instance for the particular session.

Another fundamental dependency of the SLAs, limited to the unpersonalised structures, is the connection to specific, PA-agnostic Learning Actions (LAs). For a SLA to be attained by any learner, it needs to have at least one concrete LA attached to it, which can train/teach/reinforce the particular SLA, on each of the MaTHiSiS PAs.

In terms of reference dependencies, unpersonalised SLAs encapsulate the ID of the user (usually a tutor) that has initially created them, connecting them to the collection of users taking part in the MaTHiSiS 'universe'. SLA instances are also connected to the users, in that bear the ID of the learner for which each instance applies. Is it important to note that a learner may be attached to only one unique instance of an SLA (or a unique record of runtime SLAs). Runtime SLAs in particular also encode a reference to the particular learning session where they manifested.

Figure 1 graphically illustrates the interdependencies pertaining to SLAs, based on the deployment of the MaTHiSiS database schema. The collections that the SLAs are related to are portrayed as empty placeholders for visual simplification purposes.

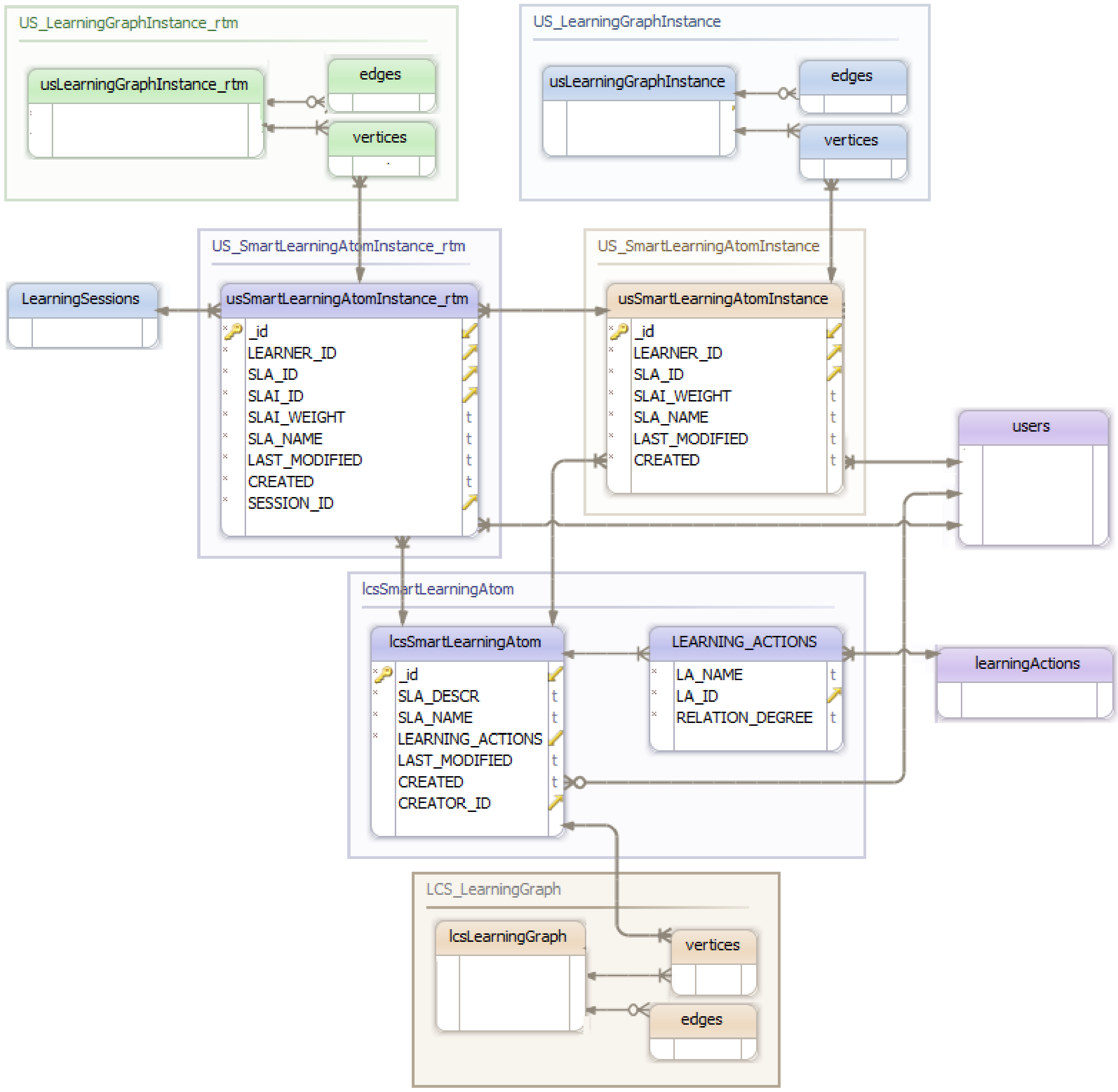


Figure 1: SLAs and SLA Instances and dependencies with other MaTHiSiS collections

2.3 Concrete Smart Learning Atoms

The following examples SLAs are created based on the Learning Graph developed for first case of ‘Age 8-9’ learners of the ASC scenario, described in Deliverable 2.2 *Full scenarios of all use cases*, linked to Learning Goals “Motor skills”, “Cognitive skills” and “Communication/Socialization skills” of the corresponding Learning Graph. This Learning Graph was elected due to the participation of its SLAs to different LGs across different scenarios for several MaTHiSiS use cases, thus exemplifying efficiently the concept of re-usability of SLAs. The particular attributes of the SLA and SLA instance data structures are detailed in Section 3.1.

2.3.1 Unpersonalised SLAs

These SLAs comprise the re-usable core SLAs, used currently in the Autism Spectrum Case (ASC) age 8-9 Learning Graph (the first of the two ‘Age 8-9’ cases in the ASC LGs), while also some (like motor sequencing, spatial and motor coordination) are used also in the Mainstream Education Case (MEC) ‘Age 8’ Learning Graph.

We can see that some SLAs share common Learning Actions, which train both skills, as in the case of LA “repeat/imitate sequence”, which is shared between SLAs “Motor Sequencing” and “Spatial Coordination”. Also, given the fact that some of the re-usable SLAs are used in both the ASC and MEC scenarios, it is worth noting that all the LAs that are attached to this SLA are included in the SLA structure, even if the Experience Engine will not consider LAs for the ASC that apply only to MEC learners, and vice versa.

SLA name	SLA Structure

Motor Sequencing	{
	"_id" : "59775ee6657e8f2f9cd4cc07",
	"SLA_DESCR" : "Details about what this SLA is about.",
	"SLA_NAME" : "motor sequencing",
	"LEARNING_ACTIONS" : [
	{
	"LA_NAME" : "repeat-imitate sequence",
	"_id" : "15825ee6657e8f2f9cd4cc07",
	"RELATION_DEGREE" : "1.0"
	},
	],
	"LAST_MODIFIED" : "2016-10-06T16:15:24.370",
	"CREATED" : "2016-10-06T16:15:24.370",
	"CREATOR_ID" : "89476cc6657e8f2f9cd4cc07"
	}

SLA name	SLA Structure

Spatial Coordination	{
	"_id" : "68525ee6657e8f2f9cd4cc07",
	"SLA_DESCR" : "Details about what this SLA is about.",
	"SLA_NAME" : "spatial coordination",
	"LEARNING_ACTIONS" : [
	{
	"LA_NAME" : "repeat-imitate sequence",
	"_id" : "15825ee6657e8f2f9cd4cc07",
	"RELATION_DEGREE" : "1.0"
	},
	{
	"LA_NAME" : "fit shapes together-jigshaw",
	"_id" : "59424e1f657e8f2f9cd4a9d3",
	"RELATION_DEGREE" : "1.0"
	},
	{
	"LA_NAME" : "build model from plan",
	"_id" : "38854a4b657e8f2f9cd4ff13",
	"RELATION_DEGREE" : "1.0"
	}
	],
	"LAST_MODIFIED" : "2016-10-06T16:15:24.150",
	"CREATED" : "2016-10-06T16:15:24.150",
	"CREATOR_ID" : "89476cc6657e8f2f9cd4cc07"
	}

SLA name	SLA Structure

Motor Coordination	{
	"_id" : "32525ee6657e8f2f9cd4cc07",
	"SLA_DESCR" : "Details about what this SLA is about.",
	"SLA_NAME" : "motor coordination",
	"LEARNING_ACTIONS" : [
	{
	"LA_NAME" : "manage motor coordination",
	"_id" : "37525bd6657e8f2f9cd4bc94",
	"RELATION_DEGREE" : "1.0"
	},
	{
	"LA_NAME" : "dance",
	"_id" : "55357af6657e8f2f9cd4c8f1",
	"RELATION_DEGREE" : "1.0"
	},
	{
	"LA_NAME" : "use body language to communicate",
	"_id" : "23389f3e657e8f2f9cd4a2c7",
	"RELATION_DEGREE" : "1.0"
	}
	],
	"LAST_MODIFIED" : "2016-10-06T16:15:24.690",
	"CREATED" : "2016-10-06T16:15:23.690",
	"CREATOR_ID" : "89476cc6657e8f2f9cd4cc07"
	}

Table 2: SLAs linked to the Learning Goal: Motor Skills

SLA name	SLA Structure

Literacy Improvement	{
	"_id" : "54525cc6657e8f2f9cd4cc07",
	"SLA_DESCR" : "Details about what this SLA is about.",
	"SLA_NAME" : "literacy improvement",
	"LEARNING_ACTIONS" : [
	{
	"LA_NAME" : "enrich vocabulary",
	"_id" : "35794ae6657e8f2f9cd4cc18",
	"RELATION_DEGREE" : "1.0"
	},
	{
	"LA_NAME" : "compose words-sentences",
	"_id" : "83855ca6657e8f2f9cd4ca32",
	"RELATION_DEGREE" : "1.0"
	},
	{
	"LA_NAME" : "combine picture-word",
	"_id" : "48965ee6657e8f2f9cd4ee21",
	"RELATION_DEGREE" : "1.0"
	}
	],
	"LAST_MODIFIED" : "2016-10-06T16:15:25.120",
	"CREATED" : "2016-10-06T16:15:25.120",
	"CREATOR_ID" : "89476cc6657e8f2f9cd4cc07"
	}

SLA name	SLA Structure

Language Comprehension	{
	"_id" : "13545bc6657e8f2f9cd4cd28",
	"SLA_DESCR" : "Details about what this SLA is about.",
	"SLA_NAME" : "language comprehension",
	"LEARNING_ACTIONS" : [
	{
	"LA_NAME" : "visualise content",
	"_id" : "39544ee6657e8f2f9cd4db36",
	"RELATION_DEGREE" : "1.0"
	},
	{
	"LA_NAME" : "predict sentence",
	"_id" : "74832ca6657e8f2f9cd4ca07",
	"RELATION_DEGREE" : "1.0"
	},
	{
	"LA_NAME" : "make connections",
	"_id" : "54765df6657e8f2f9cd4ef51",
	"RELATION_DEGREE" : "1.0"
	}
	],
	"LAST_MODIFIED" : "2016-10-06T16:15:25.330",
	"CREATED" : "2016-10-06T16:15:25.330",
	"CREATOR_ID" : "89476cc6657e8f2f9cd4cc07"
	}

Table 3: SLAs linked to the Learning Goal: Cognitive Skills

SLA name	SLA Structure

Conversation	{
	"_id" : "65873fc6657e8f2f9cd4ee13",
	"SLA_DESCR" : "Details about what this SLA is about.",
	"SLA_NAME" : "conversation",
	"LEARNING_ACTIONS" : [
	{
	"LA_NAME" : "engage in dialogue",
	"_id" : "35558bb6657e8f2f9cd4fc08",
	"RELATION_DEGREE" : "1.0"
	},
	{
	"LA_NAME" : "reply to question",
	"_id" : "22579df6657e8f2f9cd4ea44",
	"RELATION_DEGREE" : "1.0"
	}
	],
	"LAST_MODIFIED" : "2016-10-06T16:15:25.780",
	"CREATED" : "2016-10-06T16:15:25.780",
	"CREATOR_ID" : "89476cc6657e8f2f9cd4cc07"
	}

Table 4: SLAs linked to the Learning Goal: Communication/Socialization Skills

2.3.2 Personalised SLA instances

The following tables present personalised instances (for the same user) of the SLAs portrayed in the previous section.

SLA name	SLA Structure

Motor Sequencing	{
	"_id" : "95745b4f657e8f23fcb10311",
	"LEARNER_ID" : "58526b4a657e8f23fcb108aa",
	"SLA_ID" : "59775ee6657e8f2f9cd4cc07",
	"SLAI_WEIGHT" : "0.52",
	"SLA_NAME" : "motor sequencing",
	"LAST_MODIFIED" : "2016-10-06T16:38:13.490",
	"CREATED" : "2016-10-06T16:38:13.660"
	}

Spatial Coordination	{
	"_id" : "25617b4a657e8f23fcb107fb",
	"LEARNER_ID" : "58526b4a657e8f23fcb108aa",
	"SLA_ID" : "68525ee6657e8f2f9cd4cc07",
	"SLAI_WEIGHT" : "0.72",
	"SLA_NAME" : "spatial coordination",
	"LAST_MODIFIED" : "2016-10-06T16:38:11.760",
	"CREATED" : "2016-10-06T16:38:11.760"
	}

Motor Coordination	{
	"_id" : "32865e4f657e8f23fcb103d5",
	"LEARNER_ID" : "58526b4a657e8f23fcb108aa",
	"SLA_ID" : "32525ee6657e8f2f9cd4cc07",
	"SLAI_WEIGHT" : "0.65",
	"SLA_NAME" : "motor coordination",
	"LAST_MODIFIED" : "2016-10-06T16:38:13.140",
	"CREATED" : "2016-10-06T16:38:13.140"
	}

Table 5: SLA instances for user with id “58526b4a657e8f23fcb108aa”, linked to the Learning Goal: Motor Skills

SLA name	SLA Structure

Literacy Improvement	{
	"_id" : "82665b4f657e8f23fcb10351",
	"LEARNER_ID" : "58526b4a657e8f23fcb108aa",
	"SLA_ID" : "54525cc6657e8f2f9cd4cc07",
	"SLAI_WEIGHT" : "0.44",
	"SLA_NAME" : "literacy improvement",
	"LAST_MODIFIED" : "2016-10-06T16:38:12.830",
	"CREATED" : "2016-10-06T16:38:12.830"
	}

Language Comprehension	{
	"_id" : "42489b4a657e8f23fcb108f9",
	"LEARNER_ID" : "58526b4a657e8f23fcb108aa",
	"SLA_ID" : "13545bc6657e8f2f9cd4cd28",
	"SLAI_WEIGHT" : "0.31",
	"SLA_NAME" : "language comprehension",
	"LAST_MODIFIED" : "2016-10-06T16:38:12.590",
	"CREATED" : "2016-10-06T16:38:12.590"
	}

Table 6: SLA instances for user with id "58526b4a657e8f23fcb108aa", linked to the Learning Goal: Cognitive Skills

SLA name	SLA Structure

Conversation	{
	"_id" : "98876b4a657e8f23fcb108ab",
	"LEARNER_ID" : "58526b4a657e8f23fcb108aa",
	"SLA_ID" : "65873fc6657e8f2f9cd4ee13",
	"SLAI_WEIGHT" : "0.48",
	"SLA_NAME" : "conversation",
	"LAST_MODIFIED" : "2016-10-06T16:38:12.050",
	"CREATED" : "2016-10-06T16:38:12.050"
	}

Table 7: SLA instances for user with id "58526b4a657e8f23fcb108aa", linked to the Learning Goal: Communication/Socialization Skills

3. Smart Learning Atom library implementation details

3.1 Data structures

The structures of the unpersonalised SLAs, personalised SLA instances and runtime instances are detailed in the tables below. As explained in Section 2, the unpersonalised SLAs reside on the Learning Content Space (LCS) and the personalised instances (hereafter referred to SLAIs) on the User Space (US). SLAIs do not encapsulate the full structure to the core SLAs, but rather a pointer along with additional personalised information. Fields marked with (*) constitute obligatory components of the structures.

LCS_SmartLearningAtom			
Component	Description	Value	Related to
_id*	The unique DB identifier of the SLA structure	String (hex)	(US_SLAI) SLA_id
SLA_NAME*	The unique name of the SLA	String	-
SLA_DESCR	Details about what this SLA is about	String	-
CREATOR_ID*	The unique DB identifier of the user (tutor) that created this SLA	String (hex)	(Users) _id, for role: tutor
CREATED	The date and time that this SLA was first created	String (date/time)	-
LAST_ MODIFIED	The date and time that this SLA was modified last, either to rename or to attach/remove attachment with LAs	String (date/time)	-
LEARNING_ ACTIONS*	The list of PA-agnostic, generic LAs that are attached to (actuate) this SLA. There should be at least one LA attached to each SLA. This list is just a pointer to the full structure of each Learning Action.	List (String)	-
LA_NAME*	The unique name of the Learning Action	String	-
_id*	(Within the LA list:) the unique DB identifier of the Learning Action	String (hex)	(Learning Action) _id
RELEVANCE_ SCORE	(For future releases). A weight in [0,1] that denotes the relevance of the LA to the particular SLA. Default: 1.0 – in the first release	Double (as String in structure)	-

Table 8: Unpersonalised SLA data structure (LCS_SmartLearningAtom)

LCS_SmartLearningAtom			
Component	**Description**	**Value**	**Related to**
_id*	The unique DB identifier of the SLA instance	String (hex)	(US_SLAI_rtm) SLAI_id
SLA_NAME*	The unique name of the SLA	String	-
SLA_ID*	The unique DB identifier of the corresponding unpersonalised SLA	String (hex)	(LCS_SLA) _id
LEARNER_ID*	The unique DB identifier of the user (learner) that this SLAI applies to	String (hex)	(Users) _id, for role: learner
SLAI_WEIGHT	A weight in [0,1] that denotes the apprehension/uptake/competence of the particular learner towards this SLA. Initial (no previous info available): 0.3 (Opted instead of 0.0, to allow to decrease if learning process fails to engage learner from the first session)	Double (as String in structure)	-
CREATED	The date and time that this SLA was first created	String (date/time)	-
LAST_ MODIFIED	The date and time that this SLA was modified last, either to rename or to attach/remove attachment with LAs	String (date/time)	-

Table 9: Personalised SLAI data structure (US_SmartLearningAtomInstance)

LCS_SmartLearningAtom			
Component	**Description**	**Value**	**Related to**
_id*	The unique DB identifier of the SLA instance	String (hex)	-
SLA_NAME*	The unique name of the SLA	String	-
SLA_ID*	The unique DB identifier of the corresponding unpersonalised SLA	String (hex)	(LCS_SLA) _id
SLAI_ID	The unique DB identifier of the (long-term) last state of the corresponding personalised SLAI	String (hex)	(US_SLAI_rtm) SLAI_id
LEARNER_ID*	The unique DB identifier of the user (learner) that this SLAI applies to	String (hex)	(Users) _id, for role: learner

SLAI_WEIGHT	A weight in [0,1] that denotes the apprehension/uptake/competence of the particular learner towards this SLA. Initial (no previous info available): 0.3 (Opted instead of 0.0, to allow to decrease if learning process fails to engage learner from the first session)	Double (as String in structure)	-
SESSION_ID	The unique DB identifier of the learner session that this SLAI's weight was modified during runtime of the learning process	String (hex)	(Sessions) _id,
CREATED	The date and time that this SLA was first created	String (date/time)	-
LAST_ MODIFIED	The date and time that this SLA was modified last, either to rename or to attach/remove attachment with LAs	String (date/time)	-

Table 10: Personalised SLAI runtime record data structure (US_SmartLearningAtomInstance_rtm)

3.2 Functionalities

The SLA library incorporates all the methods and functionalities required to create, access and modify the SLA and SLAI data structures. In this library, long-term SLAI structures and runtime snapshots of SLAIs are treated as the same structure, applicable to different serialisation on the MaTHiSiS database through the SLA lib Open API (detailed in Section 3.3). It is implemented as a Java library which is embedded to the Open API. More specifically, the library offers functionalities to:

- Create a new SLA (Java data structure), based on input of the mandatory fields of the data structure.
- Recreate a (Java) data structure of an unpersonalised SLA, based on a given JSON input of a serialised SLA.
- Create a new SLA instance (Java data structure), based on input of the mandatory fields of the data structure.
- Recreate a (Java) data structure of an personalised SLAI, based on a given JSON input of a serialised SLA.
- For each non-mandatory field missing from the input (parameters or JSON structure), provide default values to produce a complete data structure.
    - In the case of date/time fields, the current system date and time are set, unless explicitly stated otherwise.
    - In the case of SLAIs, initial default weight (0.3) is set in the very first instantiation of an SLA to a personal SLAI, unless explicitly stated otherwise.
- Retrieve and set (update) different fields of the structures.
    - For set operations, the 'last modified' field is automatically set to the current system date and time, unless this field is explicitly declared in the input (parameters or JSON).
- For facilitating the personalisation and adaptation process, a direct functionality to update SLAI weights is exposed, which allows to set new SLAI weights without having to explicitly retrieve corresponding SLAIs to access this property.

- Similarly, for facilitating the SLA editing process, a direct functionality to insert and remove learning actions from SLAs is exposed.
- Create JSON serialisations for each of the supported data structures (SLA, SLAI) to be inserted to the MaTHiSiS DB through the SLA lib Open API.

3.3 Open API

The *JAX-RS*[1] Java API for RESTful Web Services was used to create web services according to the Representational State Transfer (REST) architectural pattern for the SLA lib Open API. Resources are connected to the Java SLA library. The library is in charge of the processing (access, creation, update) of SLA and SLAI structures and the Open API is responsible of receiving and transmitting the data to the callers. The Open API is also responsible for serialising the SLA and SLAI (and SLAI runtime) structures on the MaTHiSiS database.

Access to the SLA lib Open API is available through the central *<MaTHiSiS base URL>/api/sla/* URL pattern. MaTHiSiS components that will consume SLA lib functionalities through the Open API will be able to get data from appropriate HTTP connections (bound to specific URLs), detailed in the tables below. Parameters marked with (*) are mandatory.

URL pattern	**GET api/sla/getSLAs**		
Method	GET		
Content type	Application/JSON		
Description	Return the list of all SLAs in the MaTHiSiS repository		
Responses	If Success return a list of pointers to SLAs in JSON format		
	If Error HTTP 204 status code (No Content)		
Parameters	**Name**	**URL pattern**	**Success Response Model**

	-	-	{
			"slas": [
			{
			"sla_id": {sla_id},
			"sla_name": {sla_name},
			},
			...
			]
			}

Table 11: SLA Open API - GET api/sla/getSLAs

URL pattern	**GET api/sla/getSLA**
Method	GET
Content type	Application/JSON
Description	Return an unpersonalised SLA for a given id (and optionally, name as cross-reference)

[1] https://jax-rs-spec.java.net/

Responses	If Success return a LCS_SmartLearningAtom model in JSON format		
	If Error HTTP 204 status code (No Content)		
Parameters	**Name**	**URL pattern**	**Success Response Model**

	id*
	label
		?id={sla_id }& label={sla_name}	{
			"_id":"{SLA_id}",
			"SLA_DESCR" : "{SLA description}",
			"SLA_NAME" : "{SLA_name}",
			"LEARNING_ACTIONS" : [
			{
			"LA_NAME" : "{LA_name}",
			"_id" : "{LA_id}",
			"RELATION_DEGREE" : "1.0"
			},
			….
			],
			"LAST_MODIFIED" : "{date/time}",
			"CREATED" : "{date/time}",
			"CREATOR_ID" : "{tutor_id}"
			}

Table 12: SLA Open API - GET api/sla/getSLA

URL pattern	**GET api/sla/getLAs**		
Method	GET		
Content type	Application/JSON		
Description	Return the list of learning actions attached to the specific unpersonalised SLA		
Responses	If Success return an array of LAs in JSON format		
	If Error HTTP 204 status code (No Content)		
Parameters	**Name**	**URL pattern**	**Success Response Model**

	label
	id*
		?id={sla_id }&label={sla_name}	{
			"LEARNING_ACTIONS" : [
			{
			"LA_NAME" : "{LA_name}",
			"_id" : "{LA_id}",
			"RELATION_DEGREE" : "1.0"
			},
			….
			],
			}

Table 13: SLA Open API - GET api/sla/getLAs

URL pattern	**GET api/sla/getSLAIs**
Method	GET
Content type	Application/JSON

Description	Return the list of all SLA instances in the MaTHiSiS repository for the given learner. If a learner is not defined, return the list of all SLAIs in the repository.		
Responses	If Success return a list of pointers to SLAIs in JSON format		
	If Error HTTP 204 status code (No Content)		
Parameters	**Name**	**URL pattern**	**Success Response Model**

	uid	?uid={learner_id}	{
			"learner_id" : {learner_id},
			"slas": [
			{
			"slai_id": {slai_id},
			"sla_id": {sla_id},
			"sla_name": {sla_name},
			},
			...
			]
			}

Table 14: SLA Open API - GET api/sla/getSLAIs

URL pattern	**GET api/sla/getSLAI**		
Method	GET		
Content type	Application/JSON		
Description	Return a personalised SLAI for a particular learner, given the SLAI id (and optionally, name as cross-reference)		
Responses	If Success return a US_SmartLearningAtomInstance model in JSON format		
	If Error HTTP 204 status code (No Content)		
Parameters	**Name**	**URL pattern**	**Success Response Model**

	id*
	label
		?id={slai_id }& label={sla_name}	{
			"_id" : "{SLAI_id}",
			"LEARNER_ID" : "{LEARNER_id}",
			"SLA_ID" : "{SLA_id}",
			"SLAI_WEIGHT" :
			"{slai_weight∊[0.0,1.0]}",
			"SLA_NAME" : "{SLA_name}",
			"LAST_MODIFIED" : "{date/time}",
			"CREATED" : "{date/time}"
			}

Table 15: SLA Open API - GET api/sla/getSLAI

URL pattern	**GET api/sla/getSLAIs/rtm**
Method	GET
Content type	Application/JSON
Description	Return the list of all SLA runtime instances in the MaTHiSiS repository for the given

	learner and a given session. If a learner and session is not defined, return the list of all SLAIs in the repository.		
Responses	If Success return a list of pointers to SLAIs in JSON format		
	If Error HTTP 204 status code (No Content)		
Parameters	**Name**	**URL pattern**	**Success Response Model**

	uid
	sid
		?uid={learner_id}
		&sid={session_id}
			{
			"learner_id" : {learner_id},
			"slas": [
			{
			"session_id" : {session_id},
			"slai_rtm_id": {slai_rtm_id},
			"slai_id": {slai_id},
			"sla_name": {sla_name},
			},
			...
			]
			}

Table 16: SLA Open API - GET api/sla/getSLAIs/rtm

URL pattern	**GET api/sla/getSLAI/rtm**		
Method	GET		
Content type	Application/JSON		
Description	Return a personalised runtime record of an SLAI for a particular learner, given the session id and the SLAI id (and optionally, name as cross-reference)		
Responses	If Success return a US_SmartLearningAtomInstance_rtm model in JSON format		
	If Error HTTP 204 status code (No Content)		
Parameters	**Name**	**URL pattern**	**Success Response Model**

	id*
	label
		? id={slai_id }& label={sla_name}	{
			"_id" : "{SLAI_rtm_id}",
			"LEARNER_ID" : "{LEARNER_id}",
			"SLAI_ID" : "{SLAI_id}",
			"SLA_ID" : "{SLA_id}",
			"SLAI_WEIGHT" :
			"{slai_weight∊[0.0,1.0]}",
			"SLA_NAME" : "{SLA_name}",
			"SESSION_ID" : "{SESSION_id}",
			"LAST_MODIFIED" : "{date/time}",
			"CREATED" : "{date/time}"
			}

Table 17: SLA Open API - GET api/sla/getSLAI/rtm

URL pattern	**POST api/sla/postSLA**
Method	POST
Content	Application/JSON

type			
Description	Create or update an unpersonalised SLA on the MaTHiSiS DB under the LCS_SmartLearningAtom collection. Automatically detects SLA id from input structure. If the SLA is new, thus bears no id, it automatically creates a new structure and assigns a new unique id. Other missing non-mandatory fields of the input model are filled in with default and/or current values.		
Responses	If Success HTTP 200 status code (OK)		
	If Error HTTP 304 status code (Not Modified) If input empty HTTP 204 status code (No Content)		
Parameters	**Name**	**URL pattern**	**Input Model** **(*)Marks mandatory fields for the input to be accepted**

	-	-	{
			"_id":"{SLA_id}",
			"SLA_DESCR" : "{SLA description}",
			*"SLA_NAME" : "{SLA_name}",
			*"LEARNING_ACTIONS" : [
			{
			"LA_NAME" : "{LA_name}",
			"_id" : "{LA_id}",
			"RELATION_DEGREE" : "1.0"
			},
			….
			],
			"LAST_MODIFIED" : "{date/time}",
			"CREATED" : "{date/time}",
			*"CREATOR_ID" : "{tutor_id}"
			}

Table 18: SLA Open API - POST api/sla/postSLA

URL pattern	**POST api/sla/postSLAI**		
Method	POST		
Content type	Application/JSON		
Description	Create or update a personalised SLA instance for a given learner on the MaTHiSiS DB under the US_SmartLearningAtomInstance collection. Automatically detects SLAI id from input structure. If the SLAI is new, thus bears no id, it automatically creates a new structure and assigns a new unique id. Other missing non-mandatory fields of the input model are filled in with default and/or current values.		
Responses	If Success HTTP 200 status code (OK)		
	If Error HTTP 304 status code (Not Modified) If input empty HTTP 204 status code (No Content)		
Parameters	**Name**	**URL pattern**	**Input Model** **(*)Marks mandatory fields for the input to be accepted**

	-	-	{
			"_id" : "{SLAI_id}",
			*"LEARNER_ID" : "{LEARNER_id}",
			*"SLA_ID" : "{SLA_id}",
			"SLAI_WEIGHT" : "{slai_weight∊[0.0,1.0]}",
			*"SLA_NAME" : "{SLA_name}",
			"LAST_MODIFIED" : "{date/time}",
			"CREATED" : "{date/time}"
			}

Table 19: SLA Open API - POST api/sla/postSLAI

URL pattern	**POST api/sla/postSLAI/rtm**		
Method	POST		
Content type	Application/JSON		
Description	Create a personalised runtime SLA instance record for a given learner and session on the MaTHiSiS DB under the US_SmartLearningAtomInstance_rtm collection. Automatically detects SLAI id from input structure. Automatically produces the SLAI runtime id and inserts the session id to the serialisation of the SLAI_rtm on the MaTHiSiS DB. Missing non-mandatory fields of the input model are filled in with default and/or current values.		
Responses	If Success HTTP 200 status code (OK)		
	If Error HTTP 304 status code (Not Modified) If input empty HTTP 204 status code (No Content)		
Parameters	**Name**	**URL pattern**	**Input Model** **(*)Marks mandatory fields for the input to be accepted**

	-	-	{
			"_id" : "{SLAI_id}",
			*"LEARNER_ID" : "{LEARNER_id}",
			*"SESSION_ID" : "{SESSION_id}",
			*"SLA_ID" : "{SLA_id}",
			*"SLAI_WEIGHT" : "{slai_weight∊[0.0,1.0]}",
			*"SLA_NAME" : "{SLA_name}",
			"LAST_MODIFIED" : "{date/time}",
			"CREATED" : "{date/time}"
			}

Table 20: SLA Open API - POST api/sla/postSLAI/rtm

URL pattern	**POST api/sla/updateSLAIweight**
Method	POST
Content type	Application/JSON
Description	Update the weight of a given personalised SLA instance for a given learner without

	having to repost the entire SLAI structure. Not applicable to runtime SLAIs.		
Responses	If Success HTTP 200 status code (OK)		
	If Error HTTP 304 status code (Not Modified) If input empty HTTP 204 status code (No Content)		
Parameters	**Name**	**URL pattern**	**Input Model**
	uid* weight* id* label	?uid={learner_id}&weight={slai_weight∈[0.0,1.0]}&id={slai_id}&label={sla_name}	-

Table 21: SLA Open API - POST api/sla/updateSLAIweight

URL pattern	**DELETE api/sla/deleteSLAs**	
Method	DELETE	
Description	Delete (drop) all SLAs under the LCS_SmartLearningAtom collection in the MaTHiSiS DB	
Responses	If Success HTTP 200 status code (OK)	
	If Error HTTP 304 status code (Not Modified)	
Parameters	**Name**	**URL pattern**
	-	-

Table 22: SLA Open API - DELETE api/sla/deleteSLAs

URL pattern	**DELETE api/sla/deleteSLAIs**	
Method	DELETE	
Description	Delete (drop) all SLAIs under the US_SmartLearningAtomInstance collection in the MaTHiSiS DB. If parameter *uid* is passed delete only those occurrences that apply to this learner.	
Responses	If Success HTTP 200 status code (OK)	
	If Error HTTP 304 status code (Not Modified)	
Parameters	**Name**	**URL pattern**
	uid	?uid={learner_id}

Table 23: SLA Open API - DELETE api/sla/deleteSLAIs

URL pattern	**DELETE api/sla/deleteSLAIs/rtm**
Method	DELETE
Description	Delete (drop) all runtime SLAIs under the US_SmartLearningAtomInstance_rtm collection in the MaTHiSiS DB. If parameter *uid* is passed delete only those occurrences that apply to this learner.

Responses	If Success HTTP 200 status code (OK)	
	If Error HTTP 304 status code (Not Modified)	
Parameters	**Name**	**URL pattern**
	uid	?uid={learner_id}

Table 24: SLA Open API - DELETE api/sla/deleteSLAIs/rtm

3.4 Interface with the Front-end

Smart Learning Atoms are key elements in the pedagogical methodology introduced in MaTHiSiS, and they appear in several parts of the MaTHiSiS Front-end:

- In the Learning Content Manager (LCM), all the SLAs created by MaTHiSiS users, stored in the Learning Graphs Repository (LGR), can be browsed, viewed and edited.
- The LCM is where new SLAs can be created and then published, and also existing ones can be edited by tutors.
- Moreover, in the LCM, SLAs can also be used in Learning Graphs being created by tutors.
- Finally, both tutors and learners can examine SLAs (and related graphs) that have been involved in their Learning Experience in the Learning Experience Supervisor.

3.4.1 SLAs in the Learning Content Manager

The core functionality of the LCM is to give tutors the tools to create and edit MaTHiSiS-related content. This has been the main focus of the development for the first prototype of the LCM. Regarding SLAs, it can be broken down in four main goals:

- Provide a tool for creating and editing SLAs;
- Provide a tool for adding and connecting SLAs in a Learning Graph;
- Ensure the compatibility of these tools with the defined SLA data model;
- Establish the communication (read/write) in the LCM using the LGlib Open API for SLAs.

During the content creation process in MaTHiSiS, the different building blocks (LG, SLA, LA, LAM, LM) will not necessarily be created by the same user. Having a single, unified edition tool for all of these elements would potentially result in a tool that is too complex and not easy to use. Hence, It has been decided to create separate and independent tools for each, while still ensuring these edition tools could still be connected together to allow for a quick navigation between them when working on different elements at once.

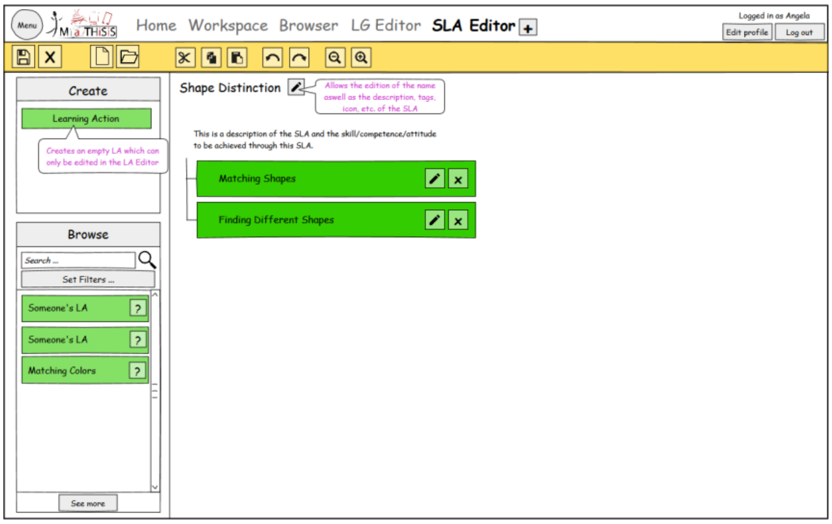
This thought process led to the definition of two additional goals for the LCM:

- Edition tools must be simple and focused on a single concept;
- Navigation between the different edition tools must be quick and easy, but not required at all times to create content.

The first step towards reaching these goals was to create UI mock-ups for the different LCM tools in order to further refine the list of functionalities required. The approach taken for designing the UI has been to work on mock-ups for all the tools (LG Editor, SLA Editor, LA Editor, LAM Editor, LM Editor) simultaneously to ensure that they all have the same look and feel and that similar functionalities in different tools would be implemented the same way in all the tools.

3.4.1.1 **SLA Editor**

Below is a screenshot of the UI mock-up for the SLA Editor tool:



Allows the edition of the name aswell as the description, tags, icon, etc. of the SLA

Creates an empty LA which can only be edited in the LA Editor

This is a description of the SLA and the skill/competence/attitude to be achieved through this SLA.

Figure 2: UI mock-up for the SLA Editor

The SLA is displayed as a list of Learning Actions, in no specific order (although the list can be rearranged at will for clarity). Each of the Learning Actions can be edited, which opens up a LA Editor in a new tab.

The user can quickly create new, empty Learning Actions from the Create panel on the left. A Learning Action created this way will have to be edited before it can be published on the MaTHiSiS cloud, but it felt important to let the user be able to completely define his SLA without having to constantly swap to a LA Editor to create new Learning Actions as s/he needs them.

The Browse panel allows the user to view and access the Learning Actions stored on the MaTHiSiS cloud, and to add existing Learning Actions (made by him/her or by other MaTHiSiS users) to his/her SLA.

Based on this mock-up, an initial prototype of the SLA Editor has been developed for the first version of the platform in the LCM native application, focusing on the core functionality: creating and editing SLAs. Existing Learning Actions cannot be browsed and used yet, but it is already possible to define the list of Learning Actions associated to an SLA and to load / save SLAs from and to the MaTHiSiS cloud, using the SLA lib Open API.

Below is a screenshot of the current state of the SLA Editor in the LCM:

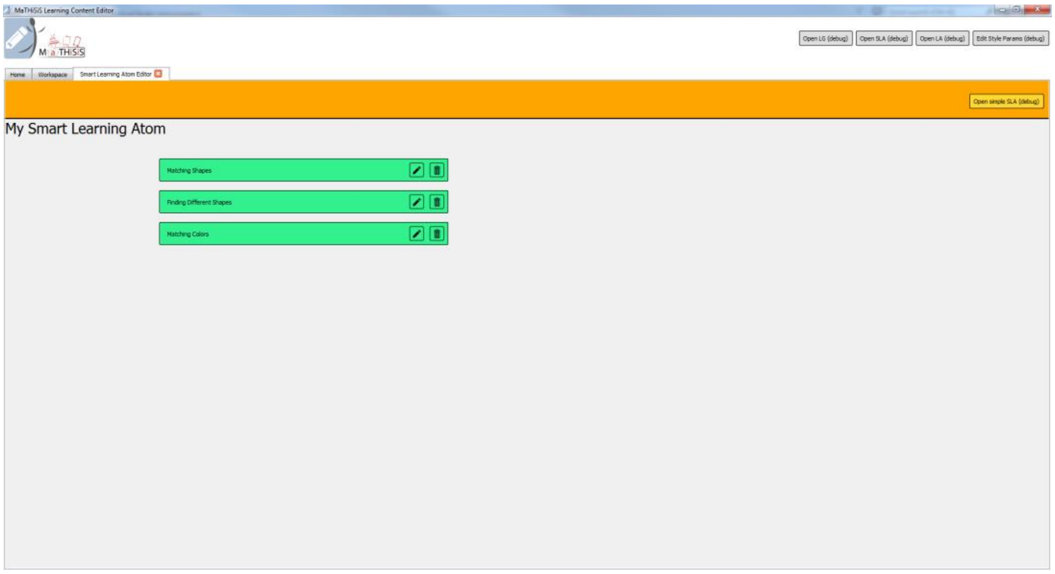


Figure 3: Current SLA Editor

3.4.1.2 LG Editor

The first prototype of the LCM also features a LG Editor tool, described in detail in D3.3 The MaTHiSiS Learning Graphs. In this tool, SLAs can be added as nodes in Learning Graphs, and connected to Learning Goals with weighted edges.

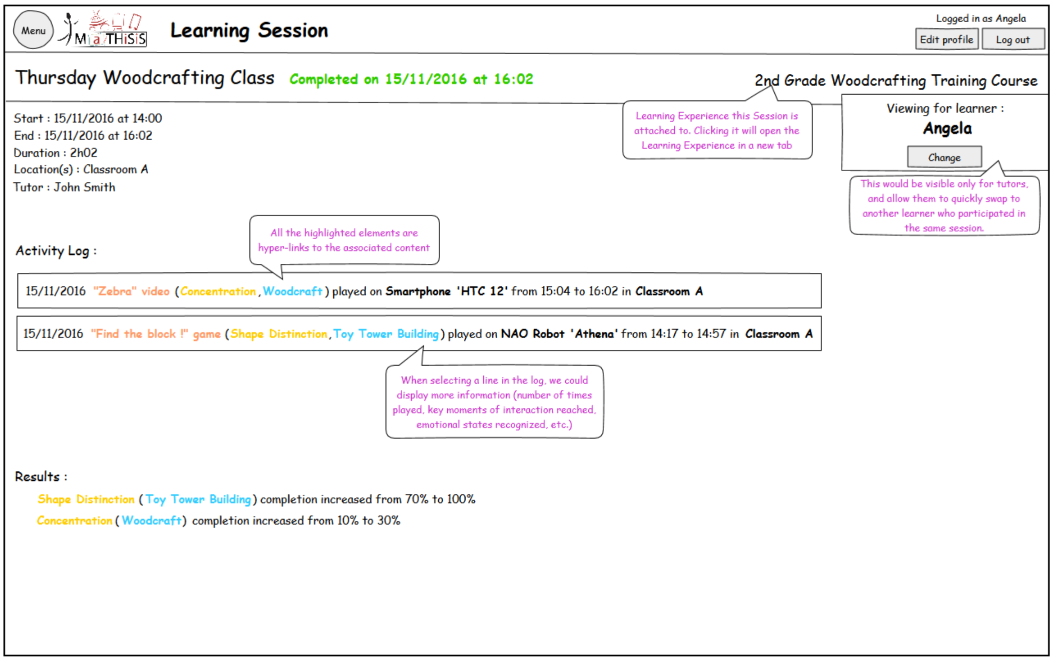
In order to fulfill the goals of this first prototype, the following functionalities have been put into place in the LG Editor:

- Create new, empty SLAs directly in the LG Editor without having to create them beforehand in a SLA Editor. This allows the user to stay focused on the graph s/he is currently editing while still being able to create all the content s/he needs.
- Open a SLA Editor to edit an existing SLA in a Learning Graph. This ensures that the navigation between the different editor tools is quick and easy, while the tools still remain independent from each other.

3.4.2 SLAs in the Learning Experience Supervisor

In the Learning Experience Supervisor (LES), both tutors and learners can review their entire Learning Experiences, down to every single Learning Session. Tutors can also manage ongoing Learning Sessions for each of their learners. SLAs that are part of the Learning Graph attached to a Learning Experience are the key, atomic elements that allow the users to keep track of what skills, knowledge or competences they've been working on.

As such, SLAs must be clearly visible throughout an entire Learning Experience. The following UI mock-up of how the LES will display information about a completed Learning Session provides a good example of SLAs (and associated Learning Goals) being highlighted for the user:



Thursday Woodcrafting Class Completed on 15/11/2016 at 16:02

2nd Grade Woodcrafting Training Course

Start : 15/11/2016 at 14:00
End : 15/11/2016 at 16:02
Duration : 2h02
Location(s) : Classroom A
Tutor : John Smith

Viewing for learner :
Angela

Learning Experience this Session is attached to. Clicking it will open the Learning Experience in a new tab

This would be visible only for tutors, and allow them to quickly swap to another learner who participated in the same session.

All the highlighted elements are hyper-links to the associated content

Activity Log :

15/11/2016 "Zebra" video (Concentration, Woodcraft) played on Smartphone 'HTC 12' from 15:04 to 16:02 in Classroom A

15/11/2016 "Find the block !" game (Shape Distinction, Toy Tower Building) played on NAO Robot 'Athena' from 14:17 to 14:57 in Classroom A

When selecting a line in the log, we could display more information (number of times played, key moments of interaction reached, emotional states recognized, etc.)

Results :

Shape Distinction (Toy Tower Building) completion increased from 70% to 100%
Concentration (Woodcraft) completion increased from 10% to 30%

Figure 4: UI mock-up for the LES

4. Conclusion

This document describes the concept of Smart Learning Atom (SLA), essential in the approach taken in the MaTHiSiS project. It allows the description of an atomic and independent knowledge or skill, which allows and will allow the users of the MaTHiSiS system to re-use their work done at this level in several learning scenarios, as it is generally done today in schools or in work environments.

In addition, this document gives details on the implementation of this concept in the first version of the MaTHiSiS platform. Even if this implementation has to evolve in the future depending on the feedback of end users, the architecture and the approach taken will remain the same. The front-end will obviously evolve substantially, since today we still lack the appropriate insight concerning concrete uses in classes or during professional training with real people involved in education and learning.

The deliverable ***D3.2 - The MaThiSiS Smart Learning Atoms*** will describe the improvements that will be made to the concept and the implementation during the second year of the project, as new versions of the platform will be set up, following the rhythm given in ***D7.1 - Integration Strategy and planning***.

5. References

[1] Nottingham Trent University (ed.): D.2.2 *Full Scenarios for all Use Cases*. Deliverable of the MATHISIS project, 2016.

[2] DIGINEXT (ed.): D2.3 *Full system architecture*. Deliverable of the MATHISIS project, 2016.

[3] Centre For Research and Technology Hellas (ed.): D3.3 *The MaTHiSiS Learning Graphs M12*. Deliverable of the MaTHiSiS project, 2017.

[4] DIGINEXT (ed.): D3.5 *Experience Engine M12*. Deliverable of the MaTHiSiS project, 2017.

[5] ATOS (ed.): D7.1 *Integration Strategy and planning*. Deliverable of the MATHISIS project, 2016.

[6] DIGINEXT (ed.): D7.2 *MaTHiSiS platform, 1st release*. Deliverable of the MATHISIS project, 2017.

[7] MaTHiSiS Description of Action, 2016.