# Creating awareness of kinaesthetic learning using the Experience API: current practices, emerging challenges, possible solutions

Maurizio Megliola[1], Gianluigi Di Vito[1],
Roberto Sanguini[2], Fridolin Wild[3], Paul Lefrere[3]

[1] Piksel Ltd., Italy
{gianluigi.divito,maurizio.megliola}@piksel.com
[2] AgustaWestland, Italy
Roberto.Sanguini@agustawestland.com
[3] The Open University, Knowledge Media Institute, Milton Keynes, UK
{fridolin.wild,paul.lefrere}@open.ac.uk

**Abstract.** We describe our use of the Experience API in preparing blue-collar workers for three frequently arising work contexts, including, for example, the requirement to perform maintenance tasks exactly as specified, consistently, quickly, and without error. We provide some theoretical underpinning for modifying and updating the API to remain useful in near-future training scenarios, such as having a shorter time allowed for kinaesthetic learning experiences than in traditional apprenticeships or training. We propose ways to involve a wide range of stakeholders in appraising the API and ensuring that any enhancements to it, or add-ons, are useful, feasible and compatible with current TEL practices and tools, such as learning-design modelling languages.

**Keywords:** Experience sharing, xAPI, verbs.

## 1 Introduction

Apprenticeship today often includes the development of 'kinaesthetic intelligence', i.e. the tactile (physical) abilities associated with using the body to create (or 'do') something involving highly coordinated and efficient body movements. Prototypical examples of this can be found in the fluid and precise motions of skilled dancers, surgeons, and skilled blue-collar workers. Gardner (2011) links high bodily-kinaesthetic intelligence to "control of one's bodily motions, the capacity to handle objects skillfully, a sense of timing, a clear sense of the goal of a physical action, along with the ability to train responses".

Becoming a skilled kinaesthetic performer through traditional apprenticeship is today largely conceptualised to take years, but this golden age of long-cycle training is quickly disappearing. Professional training environments such as in manufacturing face the challenge that the world of repetition that enabled long cycles of training to be cost-justified is increasingly taken over by short-run or personalised production,

using advanced factory machinery that does not require physical dexterity (including teachable robots that can emulate dextrous production-line workers), thereby not only causing shifts in the demand for skills, but at the same time downsizing the economically acceptable time-to-competence.

Achievements in interoperability for technology-enhanced learning (TEL) over the last decade are making it viable to develop TEL-based alternatives to a traditional apprenticeship. For example, TEL is making it possible to author, exchange, and then orchestrate the re-enactment of learning activities across distributed tools, using learning process and learning design modelling languages (Laurillard & Ljubojevic, 2011; Fuente Valentín, Pardo, & Degado Kloos, 2011; Mueller, Zimmermann, & Peters, 2010; Koper & Tattersall, 2005; Wild, Moedritscher, & Sigurdarson, 2008).

What these modelling languages fall short of, however, is the ability to handle hybrid (human-machine) experiences, to teach machines, or to train people at a distance – as required, for example, in workplaces that include robots or software agents, or in workplaces that include distributed participation across companies in a supply chain. In particular, when it comes to bridging between the virtual and the real world, between digital and physical experiences, present means are ill equipped for supporting the capturing, codification, and sharing of hybrid learning experiences.

Moreover, TEL-focused ways of capturing performance and collecting good practice are as of today still resource-intensive, placing a barrier to the spread of innovation and hindering resilience of business.

That barrier can be somewhat reduced through the use of the present version of the Experience API (ADL, 2013), and we claim could be further reduced through possible extensions and complements to the API, the need for which has become apparent in the TELLME project.

The Experience API is is a novel interface specification designed to link sensor networks together to enable the real-time collection and analysis of learning experiences conducted in different contexts, with the aim of providing better interoperability between the different types of participating educational systems and devices. Being precondition to mining and modelling, it forms a centrepiece in the canon of next generation techniques and technologies for capturing, codification, and sharing of hybrid learning experiences.

This paper presents an overview of how the components and concepts of the Experience API are integrated within the TELL-ME project to foster tracking and analysis of learning and training in three different manufacturing environments, namely Aeronautics, Furniture, and Textiles.

We describe within this contribution, how we deploy an open source Learning Recording Store to allow for collecting and reporting learning across the various components of the TELL-ME system. Moreover, starting with the Aeronautics industry, we define a taxonomy of verbs of handling and motion for capturing in particular kinaesthetic learning experiences as required in the helicopter industry, when targeting learning about a defined helicopter model and the connected standard maintenance procedures.

## 2 TELLME Learning Locker

The ADL Experience API (short 'xAPI')[1], formerly known as TinCan API, is an extension of the Activity Streams[2] specification, a format for capturing activity on social networks, created by companies like Google, Facebook, Microsoft, IBM etc., that allows for statements of experience to be delivered to and stored securely in a Learning Record Store (LRS). These statements of experience are typically learning experiences, but the API can address statements of any kind of experiences a person immerses in, both on- and offline.

While the core objects of an xAPI statement (Actor, Verb and Object) derive from the core Activity Streams specification, the Experience API has many more defined constructs for tracking information pertinent for the learner (with captured results such as "score", "success", "completion", "attempt", and "response"), unlike the Activity Streams spec which focuses on the publisher.

In its most basic application, the Experience API allows one system (the activity 'provider') to send a message (also known as the 'statement') to another system (aka the 'Learning Record Store') about something a user has done. Up until now, this process has mostly taken place inside the organisation's Learning Management System. Anything we wanted to track had to be built as part of the LMS functionality, or needed tailor-made integration. The Experience API now allows sending and receiving data between systems about what someone has done in a more openly defined way (see Figure 1).

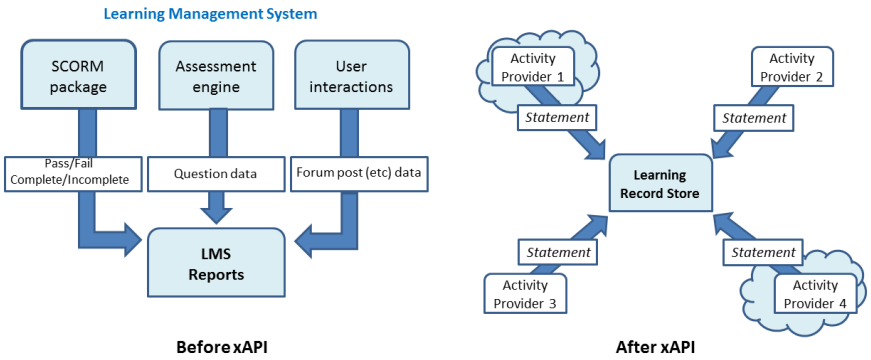


**Figure 1.** The experience tracking before and after the advent of the xAPI.

This is important, because via the Experience API, any system can send xAPI statements using a standard connection method. This process of sending xAPI statements can happen in systems behind the firewall or openly across the Internet using secure connections.

[1] http://www.adlnet.gov/tla/experience-api/
[2] http://activitystrea.ms/

Within TELL-ME, we have implemented an instance of the open-source Learning Locker[3] Learning Record Store, a LAMP-based project using MongoDB, PHP, and AngularJS, to begin collecting learning activity statements generated by xAPI compliant learning activities and reporting on such data.

The REST WS interface to the LRS was made available to be used by any component of the TELL-ME architecture to submit and retrieve xAPI statements. An example of such xAPI triple submission (with POST), using curl[4], is shown below:

```
curl -X POST --data @example.json -H "Content-Type: application/json" --
user
465ea716cebb2476fa0d8eca90c3d4f594e64b51:ccbdb91f75d61b726800313b2aa9f50
f562bad66 -H "x-experience-api-version: 1.0.0"
http://demos.polymedia.it/tellme/learninglocker/data/xAPI/statements
```

The example above is parameterized by a reference to a JSON file named example.json: it contains the actual xAPI statement in form of an actor-verb-object triple:

```
{
        "actor":{
                "objectType": "Agent",
                "name": "Gianluigi Di Vito",
                "mbox":"mailto:gianluigi.divito@piksel.com"
        },
        "verb":{
                "id":"http://activitystrea.ms/schema/1.0/watch",
                "display":{
                "en-US":"Watched"
                }
        },
        "object":{
                "id":"http://tellme-ip.eu/media/video/152"
        }
}
```

Once statements are logged, they can be queried again, for example, as needed for constraint validation to check, whether a user actually performed a certain required action step – then requiring additional constraint checker components.

[3] http://learninglocker.net/

[4] curl - command line tool for transferring data with URL syntax: http://curl.haxx.se/

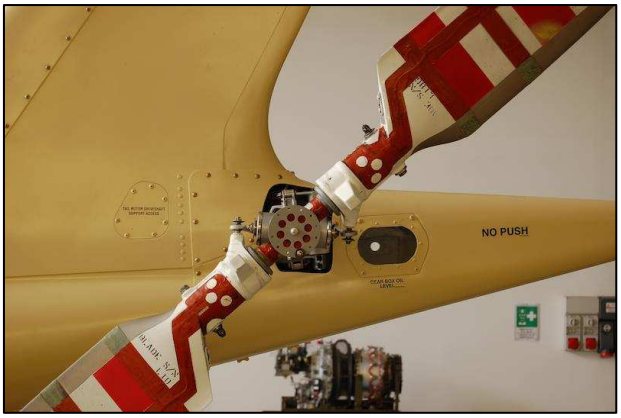


**Figure 1.** Learner interaction during maintenance of a helicopter tail rotor: 'user *cleaned* corrosion inhibitors'.

Same as any LRS implementation, the LearningLocker integrates data from multiple systems. The 'Activity Provider' can be any sort of system, where the user performs learning activity. Examples of such systems include search engines, social bookmark engines, proxy servers, social networking platforms, webinars, blogs, CRMs, as well as specific additional TELL-ME components. An example of the latter is the TELL-ME smart player application under development, which can drop statements about a user consuming a video (to the end or just the access of it). Another example is ARgh!, the platform for the delivery of context-based Augmented Reality content on mobile devices at the workplaces, which can drop statements about certain action steps being executed by the learner: Figure 1 shows an example of such user interaction with physical objects during training.

Integrating the Experience API requires not only providing an LRS, but also defining the set of 'predicates' for logging the experience statements in a meaningful way. The ADL features a standard list of commonly occurring verbs[5]. This list, however, is not bespoke to manufacturing and is restricted too much to generic, web-based learning activity, excluding interaction with objects of the real world – as required for learning by experience in manufacturing. The same is the case for ADL's extended list of "Proposed Verbs for the Experience API"[6]. Once data are mapped to the elements of the Experience API statement (see Section 3), they can be captured in the learning record store.

Figure 3 shows examples of such statements stored in the Learning Locker coming from the ARgh! trials.

[5] http://adlnet.gov/expapi/verbs/
[6] http://bit.ly/1zGmAzn

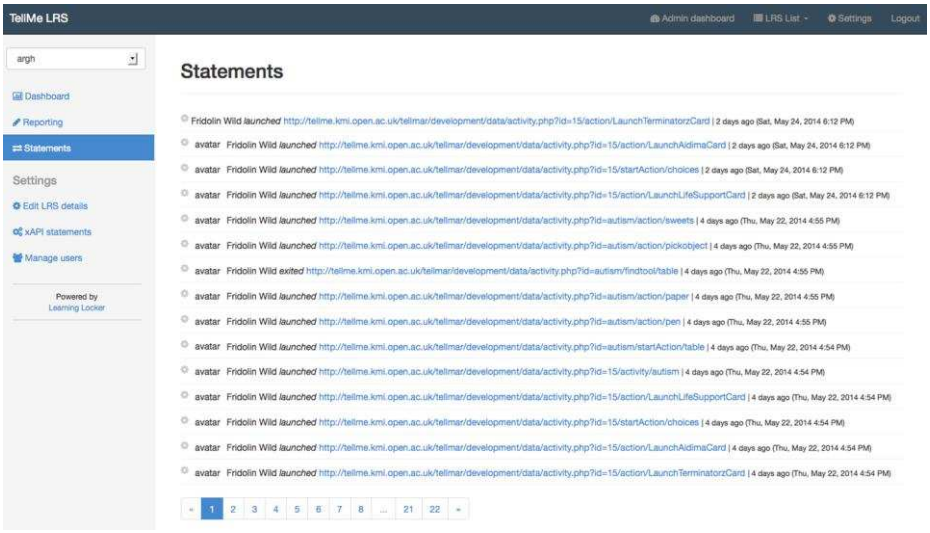


**Figure 2.** Argh! statements stored in the Learning Locker

The Learning Locker also allows creating personalized queries by means of its reporting functions, combining data from both learning activities and workplace performance, allowing linking of learning activity to performance.

One important aspect to be considered is the handling of security and the protection of the privacy of learners. In the Experience API, authentication is tied to the user, not the content. The user can be any person or thing that is asserting the statement. The user can be a learner, an instructor, or even a software agent and it can authenticate with OAuth[7], a commonly used access delegation mechanism employed by many big names such as Google, Facebook, Salesforce, etc. that eliminates the needs of sharing passwords between applications to exchange data. In this aim, the Learning Locker supports authentication and it is integrated with OAuth 2.0, exposing an API which allows 3rd parties to connect to the API via OAuth 2.0.

[7] http://oauth.net/

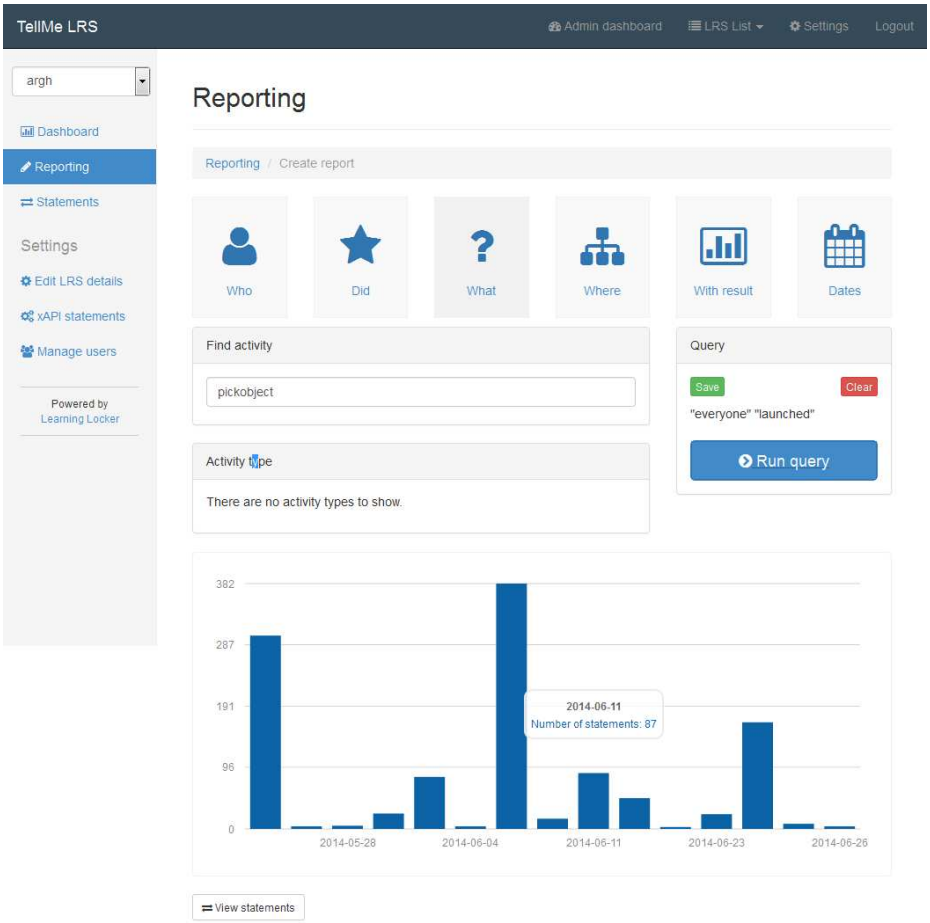


**Figure 3.** Learning Locker Reporting page

## 3 The <S,P,O> statement vocabulary (Aeronautics industry)

Each xAPI statement follows the syntax of providing a subject, predicate, and object. While subjects and objects can vary, predicates (the 'verbs') are ideally rather slowly changing and can be defined in advance.

From a cognitive linguistic perspective, there is prior work on defining verbs of motion and handling – and clarifying their relation to the way humans' cognitively process them. Roy (2005) explains how humans learn words by grounding them in perception and action. The presented theories are tested computationally by implementing them into a series of conversational robots, the latest of which - Ripley - can explain "aspects of context-dependent shifts of word meaning" that other theories fall short of. To construct a vocabulary of verbs that is widely understood, easy to learn, and natural in mapping, the work of Roy provides valuable insights: Roy postulates

that "verbs that refer to physical actions are naturally grounded in representations that encode the temporal flow of events" (p. 391). She further details that the grounding of action verbs follows the schema of specifying which force dynamics (out of a limited set) apply and which temporal Allen relations operate (p. 391). Any higher-level composition of action verbs, so Roy (p.392), can be traced back to and expressed in terms of these fundamental temporal and force relations.

This provides an angle for defining and structuring the TELL-ME taxonomy of verbs of handling and motion: it provides a basis for ordering from fundamental to composite actions, also defining their similarities. Moreover, it offers insights on how to map these verbs back to perception: it provides a rationale for how many visual overlay elements are required for an augmented reality instruction to express a certain motion verb primitive. For example, a verb 'pick up' requires the overlay visualisation of a grabbing hand as well as a highlight of the object to be picked up, whereas the motion verb 'move' consists of both 'pick up' and 'put down' actions, thus requiring more visual elements to be specified.

Palmer et al. (2005) further discuss the "criteria used to define the sets of semantic roles" for building verb classes. It provides insights into the argument structure of framesets (and so-called role-sets) of verb classes (Palmer et al., 2005, section 3).

Chatterjee (2001) reviews the cognitive relationship between language and space. He refers to Jackendoff in postulating that the "conceptual structure of verbs decomposes into primitives such as ‘movement’, ‘path’ and ‘location’ (p.57).

From an augmented reality perspective, there is additional prior work of relevance. Robertson and MacIntyre (2009) provide a review of the state of the art in displaying communicative intent in an AR-based system. The proposed taxonomy, however, stays on a level of general applicability across all sorts of augmented reality applications and is lacking the level of handling and motion required for a particular workplace, such as required in learning the maintenance of helicopters. The proposed categories (called ‘style’ strategies) are ‘include’, ‘visible’, ‘find’, ‘label’, ‘recognizable’, ‘focus’, ‘subdue’, ‘visual property’, ‘ghost’, and ‘highlight’ (p.149f). The communicative goals signified by these styling operations for visual overlays are listed as ‘show’, ‘property’, ‘state’, ‘location’, ‘reference’, ‘change’, ‘relative-location’, ‘identify’, ‘action’, ‘move’, and ‘enhancement’ (p.148). Other than the work in cognitive linguistics, here, the focus is clearly defined from a technical and not user angle: helping the user to identify or move an object is on the same level as setting labels.

In the aeronautical field, maintenance operations must be carried out according to official documents issued by the design authority of the aircraft. Such document is called the Maintenance Publication. Usually, it is organised inside the Interactive Electronic Technical Publication (IETP). The AECMA S1000D (European Association of Aerospace Constructors S1000D) is an international standard for development of IETP, utilizing a Common Source Data Base (CSDB). The standard prescribes rules to name, define, and code everything that is necessary to carry out maintenance activities in terms of:

- aircraft model;
- systems and subsystems of the aircraft;
- maintenance tasks;

- location of components;
- tools;
- additional documentations;
- miscellaneous.

Every maintenance task is identified by a unique code referring to the Task Category (e.g. 'servicing', 'repairs', 'package'), the Maintenance Activity within the category ('drain', 'fill', 'remove', 'clean', etc.), and - finally - to the definition which gives the procedure and data necessary to carry out maintenance tasks on a specific system or component. The code allows retrieving both procedures and data necessary to e.g. fill containers with fuel, oil, oxygen, nitrogen, air, water, or other fluids.

For the TELL-ME taxonomy of verbs of handling and motion the following preliminary list was compiled by pilot partners, see Table 1. The initial taxonomy presents the verbs of handling and motion as required in the helicopter industry for a defined helicopter model and the connected standard maintenance procedures. All the actions are handled by the actor 'certified staff', which therefore has not been included in the table.

| **Activity** | **Verb** | **Objects** |
|---|---|---|
| ***Operation*** | Loaded | Cargo |
| | Unloaded | Cargo |
| ***Servicing*** | Filled | Liquid or gas (fuel, oil, oxygen, nitrogen, air, water). |
| | Drained | Liquid (fuel, oil, water). |
| | Released (pressure) | Gas (oxygen, nitrogen). |
| | Lubricated | System, equipment, component, or item. |
| | Cleaned | Surface |
| | Applied (protection) | Surface |
| | Removed (ice) | Surface |
| | Adjusted | System, equipment or component. |
| | Aligned | System, equipment or component. |
| | Calibrated | System, equipment or component. |
| | Inspected (keep serviceable) | Product, system, equipment or component. |
| | Changed | Liquid (fuel, oil, water). |
| | | Gas (oxygen, nitrogen). |
| ***Examination, tests and checks*** | Examined (visual) | Product, system, equipment, component equipment, component, or item. |
| | Tested (operation / function) | System, equipment or component. |
| | Tested (function) | System, equipment or component. |
| | Tested (structure) | Structure |
| | Designed (data / | System, equipment, or component. |

| | | |
|---|---|---|
| | tolerance check) | |
| | Monitored (condition) | Product, system, equipment or component. |
| ***Disconnect, remove and disassemble procedures*** | Disconnected | Equipment, components, or items. |
| | Removed | Equipment, components, or items. |
| | Disassembled | Equipment, components, or items. |
| | Opened for access | Panels or doors (engine bay doors, landing gear doors, etc.). |
| | Unloaded (download software) | Items. |
| ***Repairs and locally make procedures and data*** | Added material | Product, equipment, component or item. |
| | Attached material | Product, equipment, component or item. |
| | Changed (mechanical strength / structure of material / surface finish of material) | Structure, surface. |
| | Removed | Material |
| | Repaired | Damaged product, system, equipment or component. |
| ***Assemble, install and connect procedures*** | Assembled | Equipment, components and items. |
| | Installed | Equipment, components and items. |
| | Connected | Equipment, components and items. |
| | Closed after access | Panels or doors (engine bay doors, landing gear doors, etc.). |
| | Loaded (upload software) | Items. |
| ***Package, handling, storage and transportation*** | Removed from (preservation material) | Products, systems equipment or components, |
| | Put (in container) | Item. |
| | Removed (from container) | Item. |
| | Kept serviceable (when in storage) | Products, systems, equipment, or components. |
| | Moved (when in storage) | Products, systems, equipment, or components. |
| | Prepared for use (after storage) | Item |

**Table 1.** Activities and their predicates (plus objects).

## 4 Conclusion and outlook

In this contribution, we have presented an overview of how the concepts of Experience API can be applied in a manufacturing environment, precisely in the Aeronautics industry. In the context of the TELL-ME project, three different taxonomies of verbs of handling and motion were prepared or are currently being prepared by the pilot partners, one of which has been described in this paper (from the helicopter industry). An instance on the open-source Learning Locker Learning Record Store was made available to any TELL-ME component, allowing them storing and retrieving xAPI statements about the workers activities.

For the future development, we intend to add preliminary taxonomies for the other two industry sectors, further mature them, and address issues related to the analytics and statements interpretation, going further ahead in understanding data and how to find meaning from it (with the help of statistical analysis). In addition, we plan to explore how to extend (keyword: 'constraint checker') or complement the Experience API to handle emerging challenges such as the need to consider the ergonomics and learning needs of human-robot workspaces. This will involve conversations and collaborations with multiple stakeholders, such as TEL vendors and users, standards bodies, and participants in robotics projects and projects involving or affected by automation – as envisioned for the Factory of the Future projects.

## Acknowledgements

The research leading to the results presented in this contribution has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no 318329 (the TELL-ME project). The authors would like to express their gratitude to the partners who have been involved in the related research work in TELL-ME.

## References

1. Gardner, H. (2011): Frames Of Mind: The Theory of Multiple Intelligences, Basic Books, first published in 1983.
2. Laurillard, D.; Ljubojevic, D. (2011): Evaluating learning designs through the formal representation of pedagogical patterns, In: Kohls & Wedekind (Eds.): Investigations of E-learning Patterns: Context Factors, Problems, and Solutions, IGI global.
3. Fuente Valentín, L. de la; Pardo, A.; Delgado Kloos, C. (2011): Generic service integration in adaptive learning experiences using IMS learning design, In: Computers & Education, 57:1160-1170.
4. Mueller, D.; Zimmermann, V.; Peters, J. (2010): ISURE: Guideline for the integration of instructional models with open educational content in IMS Learning Design, deliverable d5.2 of the iCoper consortium.
5. Koper, R.; Tattersall, C. (2005): Preface, In: Koper, Tattersall (Eds.): Learning Design: A handbook on modelling and delivering networked education and training, Springer: Berlin.

6. Wild, F.; Mödritscher, F.; Sigurdarson, S. (2008): Designing for Change: Mash-Up Personal Learning Environments, In: eLearning Papers, 9 (2008).
7. Dillenbourg, P. (2011): Design for Orchestration, In: Dillenbourg (Ed.): Trends in Orchestration: Second Research & Technology Scouting Report, deliverable d1.5 of the STELLAR consortium.
8. ADL (2013): Experience API, version 1.0.1, Advanced Distributed Learning (ADL) Initiative, U.S. Department of Defense.
9. Roy, D. (2005): Grounding words in perception and action: computational insights, In: TRENDS in Cognitive Sciences, 9 (8):389-396.
10. Palmer, M.; Gildea, D.; Kingsbury, P. (2004): The Proposition Bank: An Annotated Corpus of Semantic Roles, In: Computational Linguistics, 31(1):71-106.
11. Chatterjee, A. (2001): Language and space: some interactions, In: TRENDS in Cognitive Sciences , 5(2):55-61.
12. Robertson, C.; MacIntyre, B. (2004): Adapting to Registration Error in an Intent-Based Augmentation System, In: Ong, Nee (Eds): Virtual and Augmented Reality Applications in Manufacturing, Springer: London.