



Language Technologies for Lifelong  
Learning

LTfLL -2008-212578



## Project Deliverable Report

### Deliverable nr D3.3 – Scenario's: Threading of services

<b>Work Package</b>	WP3
<b>Task</b>	1, 2 and 3
<b>Date of delivery</b>	<b>Contractual:</b> 31-05-2010 <b>Actual:</b> 15-07-2010
<b>Code name</b>	D3.3 <b>Version:</b> 1.0      Draft <input type="checkbox"/> Final <input checked="" type="checkbox"/>
<b>Type of deliverable</b>	report
<b>Security (distribution level)</b>	PU
<b>Contributors</b>	LTfLL partners
<b>Authors (Partner)</b>	Jan Hensgens (Aurus) Jan van Bruggen (OUNL)
<b>Contact Person</b>	Jan van Bruggen (OUNL) (jan.vanbruggen@ou.nl)
<b>WP/Task responsible</b>	WP3 (OUNL)
<b>EC Project Officer</b>	Ms. M. Csap
<b>Abstract (for dissemination)</b>	Threading of LTfLL services was conceived as a means to enlarge the potential use groups of LTfLL services. This deliverable documents the design of threading of LTfLL services. It introduces levels of integration: shallow, data, workflow integration, modularisation and recombination. The deliverable gives several examples of educational usage of threading of LTfLL services. It documents in some detail a scenario for a 'long thread'. Finally the deliverable reports on a number of issues in the design of threads to be considered in the final roadmap.
<b>Keywords List</b>	Scenario-based design; SDM; service integration; modularisation; language technology; language technology tools;

LTfLL Project Coordination at: Open University of the Netherlands  
Valkenburgerweg 177, 6419 AT Heerlen, The Netherlands  
Tel: +31 45 5762901 – Fax: +31 45 5762800

## Table of Contents

1. Executive Summary .....	3
2. Introduction.....	4
3. Update of scenarios .....	6
4. Integration and threading .....	7
4.1. Background .....	7
4.2. Levels of integration and threading .....	8
4.2.1. Shallow integration .....	8
4.2.2. Data integration .....	9
4.2.3. Modularisation and recombination .....	10
5. Threads inventory .....	11
5.1. Report format and example .....	12
5.2. Examples of threads.....	15
5.2.1. Create common ground for project teams .....	15
5.2.2. Support specialisation in homogeneous groups .....	16
5.2.3. Project group composition and team building.....	17
5.2.4. Assess knowledge level of discourse for future re-use.....	18
5.2.5. Validate content of course material .....	19
5.2.6. Analyze tags to infer knowledge level.....	20
5.2.7. Support for collaborative writing.....	21
5.2.8. Peer support for question answering.....	22
5.2.9. Additional threads .....	23
5.3. Conclusions .....	24
6. Integrated thread.....	25
6.1. Team creation and team building.....	25
6.1.1. Thread Description in the LTfLL format.....	25
6.1.2. Problem Scenario .....	26
6.1.3. Solution Scenario .....	29
Actors.....	29
Data assumptions .....	29
Process description.....	29
7. Contributions to LTfLL roadmap.....	32
7.1. Introduction .....	32
7.2. Modularisation .....	32
7.3. Modularisation and recombination .....	35
7.4. Visual programming environment for recombination .....	35
8. References.....	39

## 1. Executive Summary

Workpackage 3 is responsible for the scenarios that guide the technical design and validation in the LTfLL project. The current deliverable was originally conceived as the design update for version 2 of the LTfLL services produced in workpackages 4, 5 and 6. The focus of the LTfLL project has shifted towards integration and combination of services in ‘threads’ and as a consequence version 2 of the services will be a minor update. Following these changes the scope of the current deliverable was adapted and the deliverable is dealing almost entirely with integration and threading. Updates in design scenarios will be reported in the deliverables from WPs 4, 5 and 6. Updates in validation scenarios will be reported in WP7.

Integration and threading and our approach to it are introduced in Chapter 4 where we elaborate levels of integration as *shallow integration* – as demonstrated at the annual review – *workflow integration, data integration* and *modularization and recombination*. We expect that *data integration* will be technically the most challenging, as in threads this integration will be very context-dependent and may give rise to formidable technical complexities. The approach that the project has followed in SUM architecture and widgetizing the user interface is also consistent with modularization and recombination. It is important to note that our purpose was not to create a large number of threads but to identify those that are appropriate in an educational context. In chapter 5 we present a short inventory of threads that from an educational perspective are viable. This inventory was compiled with the idea of demonstrating usability of the LTfLL tools in a wide variety of educational settings. In chapter 6 we present the candidate thread for a demonstrator of the feasibility of threading LTfLL services. The core of this thread was presented at the annual review. Here we elaborate it in more detail.

Chapter 7 summarizes some contributions to the roadmap that will be compiled by WP2 at the end of the project. The emphasis here is on modularization and recombination. We report the submodules (widgets, connector widgets and background services) that we identified and used to create the threads. A first impression is given of a visual programming environment to design and develop threads using these components as building blocks.

## 2. Introduction

The current deliverable was originally planned to contain the design update, in the form of scenarios, for the production of version 2 of the LTfLL services. The original set of scenarios was reported in Deliverable D3.2. With its appendices and annexes this had become a bulky report that contained:

- (a) the adaptation of the methodology of scenario-based design used in the project;
- (b) a first complete sets of scenarios to guide design and validation
- (c) a first evaluation of the methodology

Given the change of direction of the project away from individual services towards integrated use of the services it was decided to concentrate D3.3 on what we have called *threads* of services, rather than on delivering a new set or a fully updated set of scenarios. The deliverable therefore reports only briefly on updates in the scenarios that were thought necessary on the basis of experiences gained in development and/or the validation of the scenarios. Further work in adapting scenarios, which not only covers design of services but their validation as well, is left to the technical workpackages (4, 5 and 6) and to WP7 that guides validation.

The structure of the deliverable follows this line of action:

Chapter 3 reports *briefly* on the direction in which the scenarios of the services will develop.

Chapter 4 describes our approach to developing threads in LTfLL and the different levels of integration of the services that may be achieved. We distinguish between *shallow* integration, which is limited to integrating each individual service in a learning environment, *workflow integration*, *data integration* and, finally, *modularisation and recombination*.

Although the emphasis of the deliverable is on combining functionalities contained in the services, there are important examples of commercially available (individual) services that are offered as a training program. We discuss a few of them because they should not be neglected in our current emphasis on integration.

Chapter 5 gives several examples of threading in which functionalities of the LTfLL services are combined to provide meaningful services to educational contexts. An important consideration in the selection of a 'demonstrator thread' is the educational significance of a thread. A technically smooth operating thread does not make a valid educational case. The more complex and dedicated to a particular educational problem and a specific context the thread, the smaller the educational target group it may serve. The consortium obviously has to make final decisions on what thread to implement as part of the final development stage. To inform that decision this chapter presents several example threads.

Chapter 6 gives a detailed description of a candidate demonstrator thread that was presented in a slightly different form at the annual review.

Threading can be addressed in a modest way in the current project. The roadmap that the project will deliver at its completion has to detail the steps needed to reach the level of modularisation and recombination that we will discuss in the next chapter. To support the preparation of such a roadmap we present an inventory – from our design perspective – of modularisation of the LTfLL services, and a mock-up user environment in which users can recombine modules into new services.

### 3. Update of scenarios

The current deliverable was originally planned to contain the design update, in the form of scenarios, for the production of version 2 of the LTfLL services. Given the change of direction of the project away from individual services towards integrated use of the services it was decided to concentrate D3.3 on the so called *threads* of services, rather than on delivering a new set or a fully updated set of scenarios. To allow time to develop threads, the update of the individual services will be restricted to those changes that are required to guarantee a successful execution for the next validation round (with bigger user groups and over a longer time frame). The design updates of the services will be included in the forthcoming technical documents (D4.3, D5.3 and D6.3). All further work in adapting scenarios – which not only covers design of services but their validation as well – is left to WP4, 5 and 6 and, since it guides validation, to WP7.

Based on the previous validation data the WP7 team concluded that for WPs 5.1, 6.1 and 6.2, no changes to the problem and solution elements of the 12-page scenarios are required. However updates are required for WPs 4.1 and 4.2, and to WP5.2 following further research into amended usage scenarios.

The reorientation to threading places additional requirements on the infrastructural support for WP2. Their work will be emphasised further through constituting a possible threading and wiring environment. However, the main parts of this work will be part of the roadmap. Our initial contribution from a design perspective will be presented in chapter 7, the technical translation and decisions regarding these priorities are left to Workpackage 2.

## 4. Integration and threading

### 4.1. Background

The emphasis in the LTfLL project has shifted from individual services that are integrated in a learning environment such as Elgg to services that use each other's functionalities and data. This is reflected in the report for the interim review as well as in the deliverables D4.2, D5.2 and D6.2. In deliverable D3.2 inventories of common functionalities and resources were compiled to support the process of defining potential threads of LTfLL services. For the interim report a number of scenarios were developed in which *all* services played a role.

In the deliverables D4.2, D5.2 and D6.2, further options for integrating were presented that were dealing with integration of the services within the workpackages.

The initial scenarios were defined to demonstrate how most, if not all, of the six services could work together. It soon became clear that these all-encompassing scenarios were rather contrived. Combining functionalities of two, or maybe three services offered more realistic scenarios and were called 'threads'.

We will not discuss the process in which these ideas were developed in detail, but outline the main steps:

- 1) As a follow up to the interim review, a document was produced in which summary descriptions of the services and their educational usage were given. This document also introduced SUM diagrams for all services. The SUM diagrams of architectures already hinted at combinations of (sub)functionalities contained in the current services rather than integral services
- 2) During the Graz meeting in February 2010 the WP3 workshop was entirely devoted to define further the potential client – provider relationships between the services. The results of this workshop are contained in the threads reported in sections 4 as well as in the contributions to the roadmap reported in section 5 of this deliverable..
- 3) WP3 and WP2 collaborated to exploit the link between the analysis of common functionalities and data and the approach followed in WP2. The SUM architecture as well as widgetizing the services lend themselves neatly to an approach in which we can explore new combinations of functionalities.

In this section of the deliverable we discuss levels of integration and present an inventory – that can easily be amended by several other examples – of threads that we explicitly link to educational settings and current research in the area of educational use.

Although the emphasis of the work now is on integration of different services, there is no need to neglect the chances offered by integration of separate services. More or less supplementing the work on integration and threads, we also dwell on the way individual services can be further developed into commercially exploitable services.

## 4.2. Levels of integration and threading

In our current understanding of integration and threading we distinguish 4 levels: *Shallow* integration, (2) workflow integration, (3) data integration and (4) deep integration and recombination.

### 4.2.1. Shallow integration

With shallow integration we mean the integration of the individual LTfLL services in a learning environment such as Moodle, Ilias or Blackboard. They are accessible from within such an environment much as any other tool that may be part of such an environment. This is the level of integration that was addressed in the Description of Work for the LTfLL project. Note that the types of environments foreseen in the Description of Work are typically those of Electronic Learning Environments in which students, teachers and tutors all have their pre-designed environments and tools at hand. This level of integration was achieved by widgetizing the user interface of the services and incorporating the widgets in the Elgg environment.

Integration in a *learning environment*, whether in a VLE or a PLE, has been taken more or less for granted in the LTfLL project. Considering the business environment in which such a service is integrated, this might not be sufficient and more may be needed in one or more of the following areas:

*First*, implementation in a business environment will require that more domains and languages are supported. To a limited extent this will be achieved during the project's lifecycle. In the roadmap detailed guidelines are needed for adaptation / development / training to support additional languages.

*Second*, this type of integration will require *branding* so that the service reflects the look and feel of the service provider. A widget that allows this type of branding of the interface is already developed in the project.

*Third*, the services may require fine-tuning to context-relevant data. An example in case is the BIT specific implementation of the positioning service. Here, integration in a VLE or PLE is not the issue, but rather the selection of relevant materials as well as the definition of the training set required to tailor the system to the positioning task at hand. The standard training programs that BIT offers as well as data about prior positioning decisions can be used for this training task. Similar decisions on materials and data sets are required for other services that make use of latent semantic analysis

*Fourth*, services can be embedded ('wrapped') in a particular curriculum that might be offered with or without a learning environment. Some well-known examples of educational usage of language technology have developed in this direction. Educational Testing Service originally developed the e-rater application as a system to grade essays and to provide feedback on written English (Burstein, Kaplan, Wolff & Lu, 1996). This application was used to replace one of the human assessors for the analytical writing part of the GMAT and it is now accepted practice that an automated rating system can be used to replace one human assessor (<http://www.mba.com/mba/thegmat/gmatcoresandscorereports>). E-rater is used as scoring

engine for other tests as well. The e-rater system is now the core of a web-based writing curriculum that offers several hundreds of topics on several proficiency levels. The system provides general as well as detailed feedback on grammar, usage, mechanics and style. In much the same way applications of Latent Semantic Analysis such as the Intelligent Essay Assessor (Foltz, Gilliam & Kendall, 2000; Foltz, Laham & Landauer, 1999) and Summary Street (Kintsch et al., 2000) are now embedded in writing curricula offered by Pearson.

Several of the LTfLL services might eventually be further developed along similar lines. For example, the PenSum service may become the core of a curriculum in academic writing with an emphasis on providing feedback with respect to micro and macro coherence of a text based on one or more academic resources. PolyCAFe can be combined with chat and discussion forums to offer an environment where collaboration is supervised and students are given feedback on their interactions

#### Workflow integration

In the second level of integration we use at least two LTfLL services to offer more complex support to end-users. The level of integration is that of a workflow in which the user configures and invokes the services in a particular order. The user is responsible for configuring the services, including input and output requirements of the services. In this deliverable and in the inventory we concentrate on workflow integration, because at the level of the workflow the added value of threading scenarios becomes most obvious. For example, CONSPECT identifies concepts that are shared and unshared among learners. The results of this analysis can be used to structure a discussion using PolyCAFe and there are good reasons to have that implemented as a deliberate decision by the teacher. For instance, a teacher may create small/group teams to evoke *cognitive conflict* (Dillenbourg, 1996; Dillenbourg, 2002) by selecting as team members students with smaller overlap in concepts. Alternatively, a teacher may define a common core of concepts and select teams in such a way that students have many overlapping concepts.

#### 4.2.2. Data integration

In this level of integration the manual configuration of data for the services is replaced by automatic data integration. This level of integration was not foreseen in the DOW and any integration of this kind in the project's lifetime is more or less a bonus, because data integration between services is likely to become complex: typically, services have several ways in which they can be configured by their users. These configurations deal with inputs (such as language, domain representations in the form of ontologies and corpora et cetera) and outputs as well (such as textual feedback, grading or visualisations). Tools may output their data in particular graphical views (CONSPECT) whereas data integration (the output from one serves as input for the next) may require a much less sophisticated data set.

Automatic data integration between all LTfLL services in threads may create an environment which markedly increases in complexity. The main reason why this occurs is in the fact that the thread – that is the context of use - dictates what information or data needs to be shared. In threads several decisions have to be made regarding the input and output formats and about the way transferred data need to interpreted.

In data integration two phases should be distinguished: First, providing direct access to the same data resources and second, transferring (intermediate) data between different modules. How this transfer is handled technically (push, pull, shared temporary data storage) is part of the technical work of WP2. From the design perspective that WP3 takes, the level of data integration deserves careful reconsideration: creating too much interdependencies between LTfLL modules may result in a conflict with the service based approach .

### **4.2.3. Modularisation and recombination**

The LTfLL project aims to address a number of different problems in current higher education by using language technology. As expected, the use of language technologies in the workpackages 4, 5 and 6 shows overlap. For example: ontologies are used to represent domains in workpackages 4.1, 5 as well as 6; document corpora are used for similar purposes in workpackages 4 and 5. Functionalities to extract linguistic data from documents and discourse are used in workpackages 5 and 6 et cetera. These and other commonalities were documented in D3.2 to serve further work on integration.

Whereas integration, as discussed in previous sections, starts from the existing services, modularisation and recombination is based on an analysis of the common functionalities (and in particular overlapping functionalities). Modularisation and recombination starts with the identification of common functionalities and data across the current set of services. Examples here are the use of a common LSA engine that is used to calculate similarities between document or concept vectors. Such an inventory of commonalities was reported in D3.2. The core idea here is to go beyond that inventory and define the building blocks from which both the current as well as new services can be constructed. From a technical point of view, modularisation and recombination is based on one joint SUM architecture from which current and new services can be modelled and build.

Widgetizing a service is one step in the process of modularisation and recombination, in particular one in which user interface, objects and the interactions are defined. In addition, further data integration requires the specification of the input and output formats with possible conversions between these formats. A next step toward automation can be a definition of connector widgets to support the transfer and needed data-conversions.

## 5. Threads inventory

During the Graz workshop (February 2010) the LTfLL teams met in a ‘marketplace’ setting as sellers and buyers to bring together the demands for (additional) functionality and the functionality offered by the existing services. The major results of this workshop were:

- An overall agreement that threading goes beyond combining the set of 6 LTfLL services, and requires combinations of functionality of (sub-)modules.
- An inventory of (sub-)modules (> 20), which validated their usefulness in the first of the LTfLL services.
- A first small inventory of threads for further consideration.

Based on the project experiences, the internal discussion and analysis of the threads went into a next stage. Some of the intermediate work fulfilled important preconditions for work on threading. The description of the LTfLL services using SUM diagrams, for example, fits an orientation to modularisation. It provided more clarity for (sub-)modules and enabled us to start thinking about a LTfLL toolkit. The possibility to address and access directly LTfLL data and functionality makes the possible workflows of the threads more realistic. During the validation activities at the pilot institutions the services ran satisfactorily to meet their validation purposes (See D7.2). We consider this as evidence that supports the LTfLL services approach and it strengthens confidence that re-usable software will be produced. The choice for ELGG and Moodle as platforms to run the LTfLL services forced the development teams to adopt the widget orientation. During the review demonstrations of the software most of the services ran in this context and some of these (PolyCAFe and FLSS) used multiple widgets and inter-widget communication. The latter developments opened new opportunities for threading (e.g. a next level of integration: modularisation and recombination).

We ran a provisional analysis of the SUM diagrams to identify the most interesting sub-modules and we will use some of these in the example threads. The following table lists them with a functional name (given by WP3) and a service submodule reference to document its origins. A complete report of all sub-modules identified will be given in chapter 5.

Submodule	Service reference	Submodule	Service reference
PeerProfileMatcher	Widget 6.2-8	PhraseMatcher/Scorer	Widget 4.1-2
PositionerOnCompetence	Widget 4.1-4	AnnotationEditor	Widget 6.1-4
PhraseConceptLinker	Widget 6.1-3	TagsConceptsLinker	Widget 6.1-5
ConceptogramVisualiser	Widget 4.2-1	ConceptExtractor	Widget 6.1-2
ConceptogramComparisor	Widget 4.2-2	SemanticSearchEngine	Widget 6.2-5

**Table 1: The subset of sub-modules used for the thread examples with references to their services of origin**

This chapter reports a number of threads that we consider relevant to various educational settings. Obviously, considering the large number of threads that can potentially be created, we have to define threads on the basis of educational needs. This contributes to a methodology (including guidelines and instructions) for the final roadmap and for a directed dissemination to the developer communities. Further attention is needed for additional requirements (e.g. language, data) to design the concerning thread. In the next chapter the candidate thread for

demonstration will be elaborated with a core problem and solution scenario that illustrates our approach to threading.

Eventually, the project will prioritize the threads (for the roadmap) based on their descriptions to enable the selection of those with the best educational perspectives. Criteria for such selection are:

- Stability & performance of the (sub-)modules
- Easiness to line up the modules in one workflow
- Overlap in the learning context (Language, Domain, Instruction Strategy)
- Added value to the learning processes
- Amount of additional development required

The prioritisation process results in a short list of threads that are worthwhile to explore further. This list is input for the roadmap that will be specified at the end of the project and will be reported in D2.5. The report of the annual project review expressed a preference for the type of combinations to address in the demonstration thread. Chapter 6 gives a detailed account of that thread.

### 5.1. Report format and example

The report format is based on the methodology of Scenario Based Design (SBD) and the scenario template described in deliverable D3.1. From the template the core fields Reference, Problem, Solution, instructional Contexts, Literature and Testing opportunities were kept. The problem description is completed with a Workflow and a schematic Summary field.

Reference: .....	Workflow: .....
Problem: .....	Summary: .....
Contexts: .....	Literature: .....
Solution: .....	

**Table 2: report format for threads**

The report format uses short problem and solution descriptions and it focuses on the innovative elements<sup>1</sup> that are offered by the thread. The workflow specifies the input, the output and the data transfer in a global way. To design instructionally effective threads, a description in terms of the workflow is sufficient. If a thread is implemented at the level of workflow integration, the user remains responsible for feeding the output of one service into another or for converting the required data<sup>2</sup>. The deliverable D3.3 is a design document, which will be described in detail in D2.3 (integration infrastructure) and D4.3, D5.3 & D6.3 (versions 2 of the services). Finally, the reference field states the use of the thread, succinctly in one line. By listing the labels and summaries an easy overview can be created to enable future priority setting.

The following artificial example illustrates the thread reporting scheme. This example is situated in the formal learning environment of the University of Gulpen. The Faculty of Law uses

<sup>1</sup> We will not strive for completeness, so we neglect common interactions like logging in etc..

<sup>2</sup> The possibilities for higher levels of integration like data integration

Problem Based Learning (PBL) as a learning strategy. The masters students have an assignment to form a team of solicitors to defend the company TransFigures. The company is accused of criminal draining of toxic chemicals and of polluting the African countryside. The students should act as a multi-disciplinary expert teams that covers the different specialisations of Business Law, Environmental Law, International Criminal Law etc. Each student prepares the lines of defence from one of these perspectives and the team has to select a common strategy. The task setting, the role distribution and the accusations are provided by the teaching staff. The students themselves search background information of sufficient quality that corresponds to their different roles/perspectives. This example combines the cycles of individual and collaborative learning of the Stahl cycle.

### Example Thread Report

---

*Reference:*

Preparing individually a multi-disciplinary team to create collaboratively a common understanding and to decide a common strategy.

*Problem:*

Students have to search for background information that corresponds to the perspective to which they are allotted. They have to judge the quality and the relevance of the material found. Then they have to share that information and their understanding of it with the other students. As a team the students then collaboratively construct a common understanding and they discuss, negotiate and choose a defence strategy that combines the different perspectives.

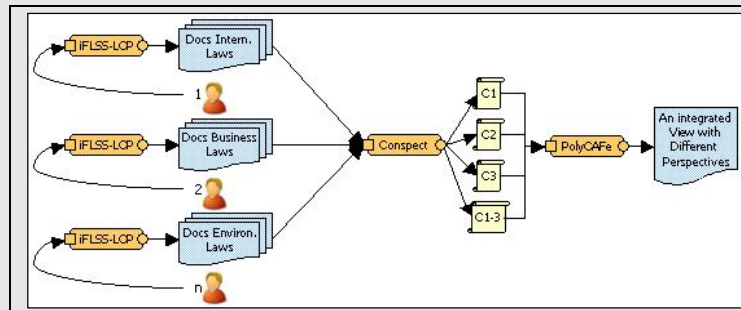
*Contexts:*

Problem Based Learning; Project Based Learning; Learning by Design;. Competence training for multi-disciplinary teams  
Law, Education; Political Sciences; Ecology

*Solution:*

We offer the students a Mash Up Personal Learning Environment (Mupple) with access to a specific subset of the LTfLL services in combination with a process worksheet. They use the iFLSS-LCP service to search the background resources. The semantic search produces better search results (quality and relevance) and gives insights in how the information is used and trusted by the member of their social network. When learners have found sufficient resources they study these to further underpin their perspectives on the case. CONSPECT is used to visualise for each perspective the concepts and their relation and to visualise the emerging group conceptogram. The visualisations are discussed using a combination of chat and forum with PolyCAFe to create a common understanding, to collect the perspectives and to negotiate the defence strategy.

*Workflow:*



For the purposes of clarification and to constitute some instructions how to read these workflow diagrams we describe each step:

- (1) Each team member uses iFLSS-LCP to collect the set of resources for their perspective. The search stops whenever resource set is of sufficient quality.
- (2) CONSPECT is used to visualise the concepts and their relations from the individual perspective as well as from the emerging group perspective.
- (3) The conceptograms are translated into the topics that need to be discussed (in chat and/or forums) to create common ground. PolyCAFe monitors the discussions and provides feedback.

*Summary:*

(prepare (search (iFLSS-LCP))) →  
 (common ground ((visualise ( CONSPECT))) → (discuss (PolyCAFe)))

*Literature:*

Clewley, G. and Bowen-Clewley, L (2005). *A Report on Multidisciplinary Approaches in Public Health*, Competency International Ltd.

*Crossfunctional teams:* <http://best.berkeley.edu/~pps/pps/teams.html> (accessed 18-5-2010).

Pfeiffer, S.I. (1981). The problems facing multidisciplinary teams: As perceived by team members. In *Psychology in the Schools*, 18, 330-333.

We add some reflections to this artificial example:

- In this example the services as used as scripted entities. In foreseen threading and in the initial inventory of threads reported in the remainder of this chapter, the sub-modules will play a more prominent role.
- LTfLL services are useful for other domains and not restricted to those currently addressed, namely Medicine and Informatics.
- For each new domain threading some additional set-ups for the services and components, such as inserting an additional domain document corpus ( CONSPECT, PenSum), an ontology (iFLSS-LCP). Moreover all all services need to share the same language.

- Even without reaching the level of data integration the thread has educational value. Data integration would make it easier to manage when the objective is exploitation as a service ‘wrapped in educational tasks’.
- In the project context we focus on workflows with the LTfLL (sub-)modules. In an educational or lifelong learning practice the workflow would probably intermingle with external tools such as Wiki or Google Doc for collaborative writing.
- The context of this thread is formal learning. However, the involvement of teachers is minimal. The same workflow can be used for informal learning or even as practice in a Law’s office.

## ***5.2.Examples of threads***

A thread, in our working definition, combines functionalities (which implies that sharing data such as a LSA-vector space is no threading) from more than one of individual services. It arranges two or more (sub)modules in a workflow. This implies that the number of possible combinations will be countless.

Most of the following examples are reported in the reporting formatted introduced earlier. The examples themselves are just a very small sample of the huge amount of educational threads that can be constructed on the basis of the same tool kit. In addition, we describe this initial set of threads only from a designer’s perspective. The intended process of prioritisation gives enough room to examine and weight the more technical considerations (e.g. is it possible to isolate the functionality for the stand-alone sub-module in time). From the perspective of the LTfLL project this is no problem at all, because the resulting list and its prioritised short list are transferred to the roadmap.

### **5.2.1. Create common ground for project teams**

*Reference:*

Create common ground for project teams

*Problem:*

In current business practice multi-disciplinary project teams are installed to manage complex problems. The strength of the contributions from different members gets lost if the team lacks common ground to enable successful communication. Students have to acquire soft skills to collaborate in (multi-disciplinary) teams. The educational institutions train these competences with project based tasks.

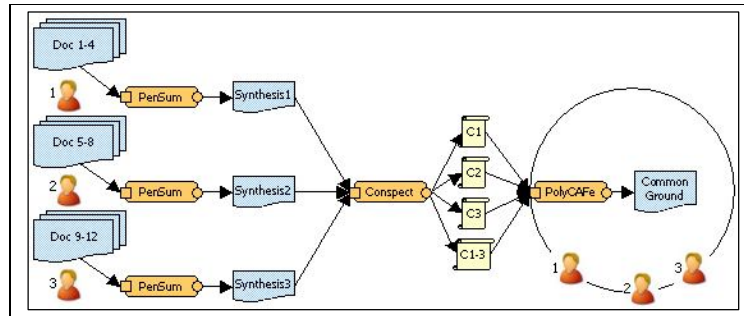
*Contexts:*

Formal: Problem/Project Based Learning virtual business learning; soft skills training;  
Law; Political Sciences; Ecology; Business and Economics

*Solution:* all students use the service PenSum to write a synthesis based on the papers allocated to them. The resulting texts are transferred into the service CONSPECT to visualize all individual and emerging group conceptograms and their relations. These results and especially

the emerging group model will be discussed using the service PolyCAFe which creates a common understanding and constitutes the common ground.

*Workflow:*



*Summary:*

(prepare (synthesize (PenSum))) → (visualise (CONSPECT))) → (discuss (PolyCAFe))

*Literature:*

Alpay, L., Giboin, A., & Dieng, R. (1998). Accidentology: an example of problem solving by multiple agents with multiple representations. In M. W. Van Someren, P. Reimann, H.P.A. Boshuizen, & T. de Jong (Eds.), *Learning with multiple representations* (pp. 152-174). Amsterdam: Pergamon.

Beers, P.J., Boshuizen, H.P.A., Kirschner, P.A., & Gijsselaers, W.H. (2005). Computer support for knowledge construction in collaborative learning environments. *Computers in Human Behavior*, 21, 623-643.

### 5.2.2. Support specialisation in homogeneous groups

*Reference:*

Support specialisation in homogeneous groups

*Problem:*

In education groups often have a large common background, because the participants are all enrolled in the same curriculum. Too much homogeneity can however be a serious obstacle for learning whenever it is required to manage, compare, contrast and bridge different perspectives, as is common practice in multi-disciplinary and collaborative team work (this problem is linked with 5.2.1).

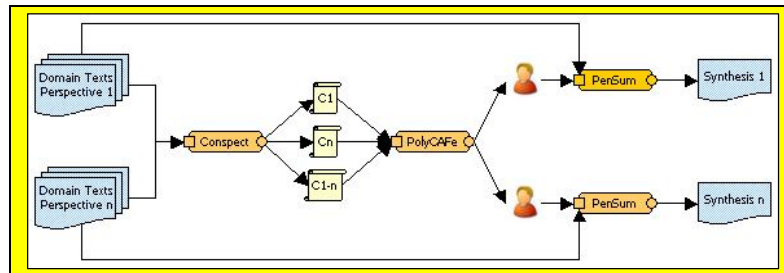
*Contexts:*

Problem/Project Based Learning, workplace training  
Informatics; Engineering; Urban Planning

*Solution:* For each perspective a collection of texts is input to CONSPECT. This service builds the different concept maps and one integrated concept map. These concept maps are input for a common discussion in PolyCAFe to help understand the different perspectives/roles and to share these among the participants. Criteria for role assignment can be prior knowledge or individual

interests. Based on the selected role all participants prepare individually their own perspective into details by using PenSum before the collaborative tasks starts.

*Workflow:*



*Summary:*

(prepare (visualise ( CONSPECT))) → (discuss (PolyCAFe)) → (synthesize (PenSum))

*Literature:*

Barkley, E.F, Cross, K.P. & Major, C.H. (2005). *Collaborative learning techniques: A handbook for college faculty*. San Francisco: Jossey-Bass.

Bosworth, K. (1994). Developing collaborative skills in college students. In K. Bosworth & S.J. Hamilton (Eds), *Collaborative learning: Underlying processes and effective techniques*. New Directions in Teaching and Learning. No. 59. San Francisco: Jossey-Bass.

Johnson, D.W., Johnson, R.T. & Smith, K.A. (1991). *Cooperative learning: Increasing college faculty instructional productivity*. ASHE-ERIC Higher Education Reports. No. 4. Washington, DC: George Washington University.

### 5.2.3. Project group composition and team building

*Reference:*

Project group composition and team building

*Problem:*

Several problems tend to be associated with project teams in education. First, it is difficult to create well-balanced project teams. How can one compose a project team that has sufficient general as well as specialized knowledge and skills that are necessary to complete project if one has to select peers who have comparable backgrounds? Second, how can we monitor and assess the quality of the team building process, which is an important success factor for project team work?

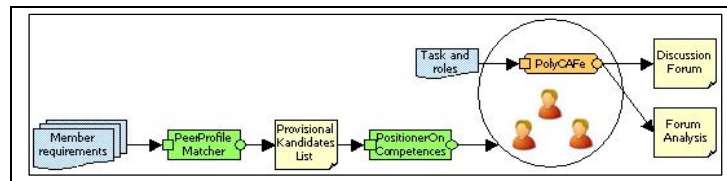
*Contexts:*

Virtual Business Learning; Project Based Learning  
Informatics; Urban Planning; Ecology

*Solution:*

The tutor uses the sub-module PeerProfileMatcher (WP6.2) to select all potential team members from the social network; then he uses the submodule PositionerOnCompetence (WP4.1) to select those with the best match of knowledge and experience. He appoints them manually to appropriate roles in the project. By using the service PolyCAFe for their ongoing discussions the team building process can be monitored and assessed by the use of indicators such as convergent language usage and a consistent focus on the main topics.

*Workflow:*



*Summary:*

(prepare ((search (PeerProfileMatcher)) → (validate (PositionerOnCompetence))))  
 → (discuss (PolyCAFe))

*Literature:*

Dong, A (2005). The latent semantic approach to studying design team communication. *Design Studies*, (26) 445-461.

Dong, A. (2004). Quantifying Coherent Thinking in Design: A Computational Linguistics Approach. In J. Gero (Ed.), *Design Computing and Cognition '04* Dordrecht, The Netherlands: Kluwer Academic Publishers.

#### 5.2.4. Assess knowledge level of discourse for future re-use

*Reference:*

Assess knowledge level of discourse for future re-use

*Problem:*

Students are frequently engaged in academic debates to build their academic skills. On-line discussion forums are one of the vehicles for these debates. For the teachers it is cognitively demanding and time-consuming to infer the knowledge level of the debate and that of the individual participants. They have to analyse the validity of the content as well as the quality of reasoning of each contribution. In addition, students often do not fully understand what exactly is expected from them during these debates and they often fail to reflect on the quality of their own contributions.

*Contexts:*

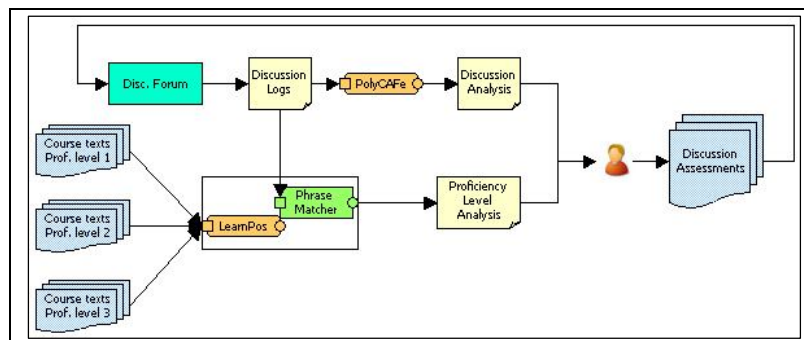
Academic skills building; Debating skills  
 Philosophy; Law; Political Sciences

*Solution:*

Prepare the service LearnPos for a specific domain, train it with discussion contributions of students of various proficiency levels (use these as standards); collect the threads of a discussion

forum use the PhraseMatcher/Scorer (WP4.1) to assess the proficiency level of the discussion content and its individual contributions. PolyCAFe will be used to analyse the discussion, in particular its structure, which will be used as input for the teacher to assess the reasoning demonstrated in the debate. Afterwards, the teacher can select the prototypical examples as illustrative and recognisable examples for next year’s students. Whenever students struggle with the requirements and expectations of a specific task they can access assessed examples of previous groups.

*Workflow:*



*Summary:*

(set-up (LearnPos)) → (discuss (discussion forum))  
 → (assess (determine level (PhraseMatcher/Scorer))  
 ↘ (analyse (PolyCAFe)))

*Literature:*

Garrison, D.R., Anderson T., & Archer W. (2000) Critical inquiry in a text-based environment: computer conferencing in higher education. *The Internet and Higher Education* 2, 87 - 105.

### 5.2.5. Validate content of course material

*Reference:*

Validate content of course material

*Problem:*

A new course text or reader has been synthesized by a teacher or a group of teachers. This new text should sufficiently cover the topics in the set of reference materials.

*Contexts:*

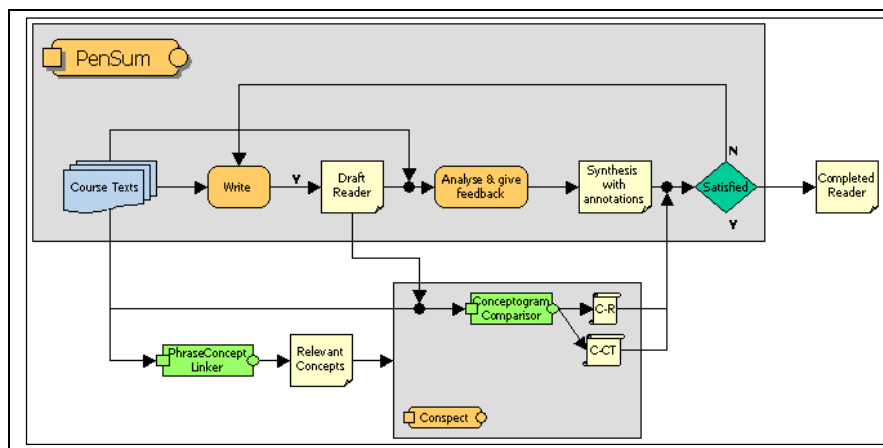
Distance Education

*Solution:*

Use PhraseConceptLinker (WP6.1) to create a list of concepts from the (automatic) annotation of concepts, tags and phrases from the reference materials. Select the most relevant concepts to be covered as norm. Use PenSum to support the different cycles of the writing process to ensure that the learning material developed is more or less similar to the reference material and that the topic coverage fits the reference material as well. During each cycle an alternative validation via

the ConceptogramComparison (WP4.2) can be done. This module generates and visualises conceptual models for the reference materials as well as the new developed material and shows commonalities and differences between them.

*Workflow:*



*Summary:*

(write (PenSum))      ➡  
 (prepare (PhraseConceptLinker))      ➡ (validate (visualise (ConceptogramComparator)))

*Literature:*

Teachout, M., Segó, D., & Ford, J. (1997). An integrated approach to summative evaluation for facilitating training course improvement. *Training Research Journal*, 3, 169-184.

### 5.2.6. Analyze tags to infer knowledge level

*Reference:*

Analyse tags to infer knowledge level

*Problem:*

Positioning of a learner is complex whenever there is insufficient text-based evidence in the learner's portfolio, or the positioning service cannot be trained to produce reliable results. The use of LSA as a means to determine expertise area and level of learners requires user produced discourse material which may be scarce.

*Contexts:*

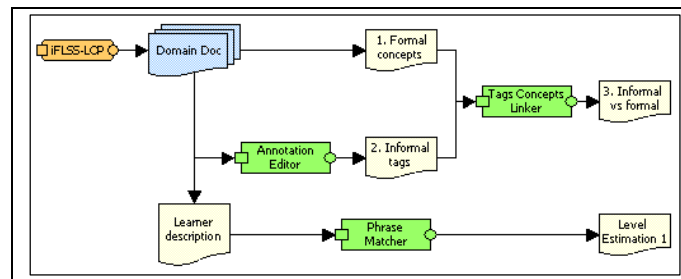
Positioning and Assessment, Assessment of Prior Learning (APL)  
 Engineering; Arts

*Solution:*

In addition to having users produce discourse, input texts or an up-to-date portfolio, they may be engaged in categorization and description tasks in which they tag and describe material that is

available in FLSS-CES (WP6.1): using the AnnotationEditor (WP6.1) and a text editor. The descriptions are fed into the PhraseMatcher/Scorer (WP4.1) and the learner tags are analysed by the TagsConceptsLinker (WP6.2). Two indicators for the proficiency level are computed. The first is based on the assumption that the used phrases can be compared to ‘golden standard phrases’ of the different proficiency levels. The second is based on the assumption that growth of expertise will manifest itself in tagging that will get closer to a formal ontology and that tags will become more abstract, encapsulated et cetera. The comparison between the informal tags and the formal concepts can be used to indicate the proficiency of the learner in particular domains. The approach followed here closely mimics categorization of problems (Chi, Feltovich, Glaser) which is a task well-known to discriminate between beginners and experts. Both indicators are reported to the teacher who makes a final assessment.

*Workflow:*



*Summary:*

(prepare (select(iFLSS-LCP))) → (describe(Free choice)) → (validate(PhraseMatcher/Scorer))  
 ↘ (tag(AnnotationEditor)) → (validate(TagCoceptLinker))

*Literature:*

Chi, M.T.H., Feltovich, P.J., & Glaser, R. (1981). Categorization and representation of physics problems by experts and novices. *Cognitive Science*, 5, 121–152.

Nievelstein, F., Van Gog, T., Boshuizen, H.P.A., & Prins, F.J. (2008). Expertise-related differences in ontological and conceptual knowledge in the legal domain. *European Journal of Cognitive Psychology*, 20, 1043-1064.

### 5.2.7. Support for collaborative writing

*Reference:*

Support for collaborative writing

*Problem:*

Environments that offer collaborative writing, such as Google docs or wikis only offer bare co-authoring of documents. Current CSCL environments offer a *combination* of co-authoring and discussion or argumentation spaces. However, there are no environments in which feedback is given on the content as well as on the discussions between authors.

*Contexts:*

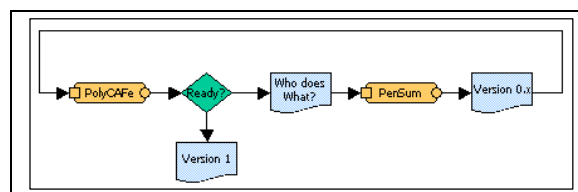
Collaborative paper writing, active & explorative learning.

Law; Political Sciences

*Solution:*

The services PolyCAFe and PenSum can be combined to exploit the complementary nature of their functionalities. This combination supports writing and discussion processes by providing relevant feedback on content of writing and the quality of the discussion by making it transparent and inspectable. The optimal use of PenSum will be mainly confined to well-defined writing assignments, which requires a carefully prepared environment. The current combination transfers the direction and guidance support more to the peer group thereby opening opportunities for less-defined writing tasks.

*Workflow:*



*Summary:*

(repeat (discuss (plan(PolyCAFe))) → (write(PenSum))) → (validate(PolyCAFe))

*Literature:*

Stahl, G. (2006), *Group Cognition: Computer support for building collaborative knowledge*. Cambridge, MIT Press.

Baker, M. (2003). Computer-mediated argumentative interactions for the co-elaboration of scientific notions. In: Andriessen, J., Baker, M., Suthers, D. (Eds.), *Confronting cognitions: Arguing to learn*. Kluwer Academic Publishers, Dordrecht, The Netherlands, pp. 47-78.

### 5.2.8. Peer support for question answering

*Reference:*

Peer support for questioning answering

*Problem:*

Tutor load is increasing, especially when tutors are required to provide in-time responses or support to student questions. Peer support can be used to handle these questions *if* competent peers can be identified and allocated to this support activity. Current implementations of peer support limit the scope of potential question-answerers to those currently enrolled in the same course that the help seeker is in. This reduces the chance of finding suitable peers.

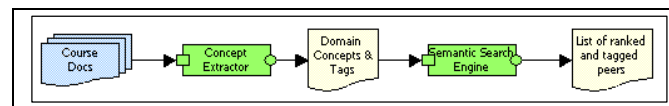
*Contexts:*

Distance Education; competence-based learning, workplace learning  
Medicine;

*Solution:*

The automatic ConceptExtractor (WP6.1) is used to map the course content to topics in the domain ontology and/or tags. With the generated concepts and tags competent peers are located through the SemanticSearchEngine (WP6.2). Whenever a learner puts forward a question or a request for support, the context from which the question originated will be used to infer the relevant topics to which peers are related. The learner will be invited to contact the peers that the system has found. This initial thread is mainly to prepare the virtual peer group. Future improvements can be found in scheduling and more detailed question analysis.

*Workflow:*



*Summary:*

(prepare (annotate (ConceptExtractor)) → (search(SemanticSearchEngine)))

*Literature:*

- De Bakker, G., Van Bruggen, J., Sloep, P., Jochems, W. (in press). Introducing the SAPS model and a corresponding allocation mechanism for synchronous online reciprocal peer support activities. *Journal of Artificial Societies and Social Simulation*.
- Van Rosmalen, P., Sloep, P., Kester, L., Brouns, F., De Croock, M., Pannekeet, K. & Koper, R. (2008). A learner support model based on peer tutor selection. *Journal of Computer Assisted Learning*, 24, 74-86.
- Van Rosmalen, P., Sloep, P., Brouns, F., Kester, L., Kone, M., and Koper, R. (2006). Knowledge matchmaking in Learning Networks. *British Journal of Educational Technology*, 37, 881-895.

### 5.2.9. Additional threads

In the preceding paragraphs we described a first set of threads. This set only serves the purpose to give an overview of possible approaches and to exemplify the ideas behind threading. We conclude these descriptions with some additional ideas, reported in shorthand rather than using the reporting format, about the use of the LTfLL toolkit for solving educational problems.

- The FLSS (WP6.1) services can create metadata for the increasing number of open educational resources that are not yet indexed in a systematic manner.
- The TopicExtractor (WP6.1) can be used as preprocessor to filter terms that will be used for monitoring the conceptual development with CONSPECT (WP4.2).
- The (i)FLSS services can be used to provide an open (not predefined) access to additional texts and resources for concepts that are missing or less well understood from within PenSum.
- The knowledge rich approach of the 4.1 service requires annotated documents. A (semi-) annotation process of the FLSS service could prevent the tedious handwork of annotating.
- The CONSPECT conceptogram visualiser could be used by the learners of PenSum to draw concept maps and to locate main concepts from their reading materials as preparation for the synthesis writing.

- The (i)FLSS services can extend the feedback given by the other services by providing access to suitable additional resources.

### 5.3. Conclusions

Threading can be constructed at two levels by using the individual services or the modules of the services. In the current status of the LTfLL project these options are not always available. Although we are still in the early phase of the process of modularisation and encapsulating relevant functionality, on the basis of current development, we may expect that using the modules as entities will produce leaner threads or new services. Furthermore, from the examples we have shown, threading offers opportunities to address other educational problems and that additional services can be created. These new development efforts exceed the resources and scope of the LTfLL project and so they will become part of the roadmap.

In developing individual services as well as threads, there should be a balance between being generic and being specific. To find real user groups a certain level of *specificity* is required. Specific use of services and threads is supported by the SBD methodology and the scenarios in which the stake-holder perspective is represented. As an example, the focus of the current approach to learner positioning is determined by the context of Bit-Media. However too much specificity can be an obstacle for additional user groups, so the core of each service and thread should be generic but configurable and potential user groups may be extended and enhanced by exploring transferability of the services and threads.

The concept of threading enables ad hoc adjustments to specific stake-holders and specialisations for specific educational purposes, which increases the exploitation potential. However each thread itself will have a niche market with a limited user group. In addition threads containing more services and modules in their workflow will increase the complexity for their users.

The description of the threads and our experiences show that these threads can be categorised according to certain workflow patterns. We have seen the following patterns:

1. Sequential e.g. with logical time lines (preparation – execution – evaluation, see 5.2.3)
2. Parallel use e.g. to improve the (intermediate) results (see 5.2.5)
3. Mirroring e.g. creating common ground vs specialisation (the reader may try to interpret thread 5.2.1 when it is read from right to left)
4. Cycling e.g. to improve the results for each iteration (see 5.2.7)

The service PolyCAFe and threads like 5.2.1 can support different phases of a project or collaboration lifecycle. Even if the technical configuration and the domain specific preparation is exactly the same, the context of use can be very different. For example, a converging discussion to create or maintain common ground could be replaced by a diverging approach (brainstorming) to explore the domain or to get insights into different options for the project methods and instruments to be selected (in the SBD methodology different scenarios can be designed around the same thread)

While most threads are specific some (e.g. 5.2.2 and 5.2.8) can be so generic that they may become exploitable as additional services. In that sense these will behave like the original services as described in chapter 4. They can be combined into new threads or encapsulated in specific curricula.

## 6. Integrated thread

In the final year of the LTfLL project the main focus will be on the threading of services. This chapter describes in some detail the educational scenario that will be used to show the integrated use of most of the LTfLL services. This scenario (long thread) matches the requests of the reviewers, the integration of services WP 4.1 and WP6.1 (short thread) will be presented in the forthcoming technical deliverables. We specify the design of the long thread using the LTfLL format for thread descriptions. Its global setting will be provided in the sections 6.2 and 6.3, where a problem and a solution scenario for the integrated thread are introduced.

Adhering to the methodology adopted by the project, i.e. the Scenario Based Design (Rosson & Carroll, 2002) we describe the thread in terms of the problem and solution scenario used earlier in the project. Whereas the design of the services has completed one cycle – there are complete sets of scenarios – the thread(s) are in their first cycle, and correspondingly we will only present conceptual design in the form of an (abbreviated) problem and solution scenario.

### 6.1. Team creation and team building

#### 6.1.1. Thread Description in the LTfLL format

*Reference:*

Team creation and team building

*Problem:*

Successful teams need persons who are able to play different roles and who offer specific skills and expertise to the team. However, to profit from the diversity of skills and experience team members need communication skills and the team has to develop a common understanding of the problem and the approaches to be taken. In companies these specialisations are available, but in education this is quite often missing.

*Contexts:*

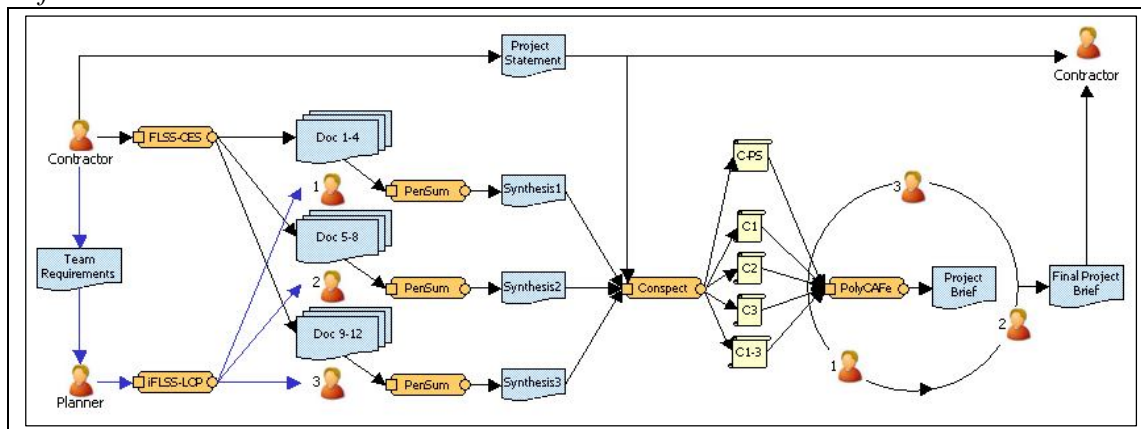
Problem / Project Based Learning, virtual business learning  
Informatics: PUB-NCIT; OUNL; Medicine: UniMan

*Solution:*

In education projects can be mimicked (e.g. by running a virtual company): the role of the problem owner (principal) is taken by the teacher. One tutor takes the role of a contractor. She prepares the project statement, specifies the requirements and lists the specific specializations with the additional roles. She uses the FLSS-CES to collect a set of background documents for each specialization and instructs the planner to search for the best possible match of the students

needed using the iFLSS-LCP. The different roles are allocated and all learners start to use PenSum to prepare their role by studying the document set of the specialization. The project statement and the syntheses are used as input for Conspect and the discussions monitored by PolyCAFe start with the different conceptograms. The produced visualisations of common and specialised concepts and the communality with the project statement create a solid base for the Project Brief that has to be written next. The project brief is written collaboratively and discussed in PolyCAFe until the team considers it to be ready. The project brief is handed over to the Contractor for final validation..

*Workflow:*



*Summary:*

(prepare (select documents (FLSS-CES)) + (select members (iFLSS-LCP)))  
 → (N-times (synthesize (PenSUM)) → (visualize (Conspect)) → (discuss (PolyCAFe)))

*Literature:*

Dong, A. (2005). The latent semantic approach to studying design team communication. *Design Studies*, 26, 445-461

Dong, A. (2004). Quantifying Coherent Thinking in Design: A Computational Linguistics Approach. In J. Gero (Ed.), *Design Computing and Cognition '04* Dordrecht, The Netherlands: Kluwer Academic Publishers.

Clewley, G. and Bowen-Clewley, L. (2005). A Report on Multidisciplinary Approaches in Public Health, Competency International Ltd.

Crossfunctional teams: <http://best.berkeley.edu/~pps/pps/teams.html> (accessed 18-5-2010).

Pfeiffer, S.I. (1981). The problems facing multidisciplinary teams: As perceived by team members. *Psychology in the Schools*, 18, 330-333.

**6.1.2. Problem Scenario**

Project teams are installed to manage complex problems, both in business and educational contexts. The project team brings together persons that play different roles and offer specialized skills and expertise to the team. To profit from the diversity of skills and experience a team has to develop a common understanding of the problem and the approach that the team will follow to tackle the problem it has to solve.

In a company setting team composition is steered by what Simon (1996) called ‘satisficing’: compose a team in such a way that there is sufficient (but not too much) expertise to reach an acceptable solution in limited time. In educational settings team or project based work can be the guiding principle that organizes large parts of or even the complete curriculum; examples in case are Problem Based Learning – often found in medical curricula –and Learning by Design. Often teams in these curricula are rather homogeneous and there is little differentiation of roles and expertise in the team. Creation and maintenance of common ground is then normally left as a form of “unsupervised learning” to the project team. Whenever team formation and roles mimic more closely a company setting, both selection of team members and ensuring common understanding become issues that need more attention. First, selection of team members is more complex, because it is not based on existing expertise but on expertise that can be developed to a level sufficient for the project. Second, more attention is needed in guiding and monitoring the proficiency of the team members and their common understanding of the problem.

### *Demarcation*

The scenario given here describes a situation in higher education, where a group of students is acting as a project team. In our example students and staff are located in an Informatics department. Students bring different expertise to the team: there may be specialists in topics such as project management; (conceptual) modeling; systems design; software development; implementation et cetera.

The scenario given here is limited to the initial phases of a project, namely those of the formation and preparation of a project team and the development of common understanding of the problem that becomes part of the project brief.

The scenario described here is based on the assumption that the *expertise needed to tackle the problem can be identified in advance*. We are aware of the limitations of this assumption: It has been argued that this approach does not fit ‘wicked problems’ (Rittel & Webber, 1984) where stakeholder issues and representations are the prime concern (think of problems such as deciding where to build a nuclear site, that are referred to as NIMBY (not in my backyard). Attempts to support such problem solving are presented by Conklin (Conklin, 2002; Conklin & Weil, 1997) and (Selvin, 2003) who suggest to structure discussions in an issue based manner that better fit an initial project phase in which all views and stakes are to be presented .

The current scenario is *not meant* to support a project phase in which the team collects a large number of issues for later consideration.

### *Analysis*

Problem solving proceeds in a number of phases that put distinct requirements on representation and means of communication (Van Bruggen, Boshuizen & Kirschner, 2003). Examples of how interdisciplinary teams build common ground are given by Alpay and Giboin (1998) who elaborate on common representations and procedures in their study of French teams who study traffic incidents. In such teams expertise from areas such as car engineering, traffic guidance and psychology is brought together. Alpay and Giboin demonstrate how the team develops a limited joint model for traffic accidents that allows the team members to maintain common ground.

Direct interventions to foster the creation of common ground are scarce: Beers et al (2006) used

explicit cues and functionalities to stimulate the negotiation of common ground with limited results. Duffy et al. (Duffy, Dueber & Hawley, 1998; Sloffer, Dueber & Duffy, 1999) used different discussion forums with different structuring (issues, topics, decision makings) to structure the problem solving process of PBL teams, but this admittedly lacked navigational support. An approach that emphasizes the process nature of the creation and maintenance of common ground is to monitor ongoing discussions in a project team. Dong (2004) showed that in successful project teams discussions were converging – as evidenced by the dimensionality in the LSA analysis of their discussions.

### **An educational example: Virtual Business Learning**

The virtual company scenario elaborated here is located between formal education (PBL) and enterprise settings. Students in Informatics at the Open University of the Netherlands are engaged in a study activity called “Virtual Company”. Virtual Company is a scenario (Westera & Sloep, 1998) that engages learners in a real life situation, modeled as a commercial firm. At OUNL the environmental consultancy firm is running for over a decade now.

Virtual companies are modeled after real firms:

- They deal with real customizers
- Projects are originated and contracted by real customers
- They have their own set of business rules documented in the firm’s archive
- Students are treated as regular personnel; they have to perform in the projects they are appointed to; they are subjected to forms of assessment that correspond to enterprise habits rather than to academic achievement assessments

In virtual companies projects typically strive for two sets of outcomes:

- the project’s final deliverable in the form of an advice, an analysis of a customer problem; a system design, or even a programmed (prototype) system.
- The individual performance improvements of the students in content areas in Informatics as well as soft skills (project management, writing, presenting et cetera).

Although virtual companies are modeled as real firms there is one important difference: virtual company is designed with the goal to improve the students’ knowledge and skills. Therefore, team composition is not optimized to reach the project’s goals as efficiently as possible, but rather to compose the team in such a way that those who are not completely fit to handle a task in the project are allotted to it.

The projects in the virtual company are run by project teams that bring together different soft skills (experience in project management; communication skills et cetera) as well as specialist knowledge. A project will involve a number of steps in system development and thereby requiring knowledge and skills in areas such as business process analysis and modeling; systems analysis and design and software development methods; implementation and evaluation strategies et cetera.

### 6.1.3. Solution Scenario

The solution described here uses functionalities from a number of LTfLL services to provide a solution to the problem of team formation and assisting the team in developing common ground.

#### Actors

In the scenario we distinguish a number of roles. Especially in educational contexts all roles may be enacted by a single teacher who designs the problem; composes the team and manages their process. The more the curriculum approaches Virtual Business Learning the more these roles will be enacted by different persons.

*Principal* – the owner of the problem. In our scenario this can still be a contracting partner (think of the Virtual Business example) but most importantly the principal presents a real-world problem case that needs to be tackled.

*Contractor* – actor responsible a first analysis of the problem in which a first estimate is made of the expertise and resource allocations that are needed to solve the problem.

*Resource planner* – actor responsible for selection and allocation of persons (learners) to the project team and possibly to project tasks (not further detailed in this description).

*Project manager* – leads the project team; responsible for the production of the project brief that captures the team’s understanding of the problem and its planned approach to a solution.

*Team member* – person allocated to the team on the basis of the particular expertise that the person can bring to the project. In educational settings the team allocation is based on expertise that can be further developed in the project rather than on existing proficiency.

#### Data assumptions

The scenario assumes a number of data resources:

1. Documents to represent domain and corporate knowledge (in LSA space) – “corporate memory”
2. Domain ontology to be inspected by formal and informal learning system and PolyCafe
3. NLP pipe
4. Qualification description used to allocate roles in the project
5. All students of the department are members of and maintain their profile / portfolio in Facebook.
6. All items 1 to 5 share at least one common language

#### Process description

*Phase 1: Analysis and team formation*

#### Step 1.

The Contractor analyzes the project description and identifies the types and levels of expertise needed in the team. The Contractor estimates the amount of resource needed. This analysis is passed on to the Resource Allocator.

#### Step 2.

The Contractor navigates the domain ontology using FLSS to locate course material that matches the types and levels of expertise. The Contractor identifies the documents that most closely match the identified type and level of expertise.

#### Step 3.

The Resource Manager navigates the ontology for the different types and levels of expertise. Using the iFLSS the Resource Manager identifies team members whose profiles match the expertise needed for the team. The Resource Manager selects team members and the Project Manager together with the Contractor.

#### Step 4

The Contractor specifies the Project Statement.

The Contractor identifies the documents to be used as 'golden standard'

The Contractor identifies the course materials that *all* team members must study to qualify for the team

The Contractor selects the course material that *specialist* team members must study to qualify for the team.

The Contractor identifies additional reading for the specialist team members

The Contractor identifies *core concepts* for the common and the specialist areas.

The Contractor prepares PenSum and allocates team members to their PenSum environment

#### *Phase 2: Establish sufficient proficiency*

The Project Manager invites the team members to join the discussion of common understanding. In this phase the team members individually use PenSum to establish common and individual expertise at a sufficient proficiency level. PenSum is used in two rounds to establish the expertise. In round one all members work (individually) on the common course materials and their associated additional readings; in round two each team members works on the specialist course materials and the associated additional materials.

- (1) all team members use PenSum to prepare a synthesis of the joint documents
- (2) each team member uses PenSum to prepare a synthesis of the specialist documents

The way of operation is the same: team members prepare syntheses of the documents studied and PenSum presents feedback and additional reading until the syntheses produced sufficiently match the standards that were assigned by the Contractor.

#### *Phase 3: Establish common understanding*

In this phase team members are engaged in a joint discussion to establish common understanding. To steer that discussion the results of the previous phase are analyzed using ConSpect..

- (a) Common documents – ConSpect analyzes the common documents using the list of core concepts specified by the Contractor and it signals overlaps among and differences between the team members
- (b) Specialist documents – ConSpect analyzes the specialist documents using the list of core concepts specified by the Contractor and it signals overlaps among the team members..
- (c) ConSpect compares all documents to the project statement using the concepts used in the analyses (a) and (b).

The Project Manager takes the output of ConSpect to structure the following discussion. The results of a) and b) identify common topics that may need additional discussion, as well as common topics that emerge from the different specialist views. The analysis of c) – which obviously has the weakest empirical backing – is used as a check whether all concepts are handled indeed.

The Project Manager prepares a discussion (chat / forum) by identifying the domain (the ontology) and the topics that need to be discussed. PolyCafe provides feedback on the level of the discourse both to the members as well as to the Project Manager who monitors and manages the dialogue.

After this discussion round, a project brief is prepared collaboratively by the team. The brief expresses both common understanding as well as individual contributions of the team members. The Contractor validates the final project brief for approval.

## 7. Contributions to LTfLL roadmap

### 7.1. Introduction

At the time the LTfLL project was launched it was clear that it would be a R&D project with the focus on the Research efforts. The methodology chosen was iterative design and the first two iterations were planned during the lifetime of the project. The development would not stop at the end of the project, but major dissemination and exploitation efforts would be made to make other developers and educational practitioners in the TEL communities aware of the results. Today, we still expect to elaborate the next development cycles together with them. The software created, lessons learnt and the requirements and new ideas will be shared in the LTfLL roadmap. In this roadmap the experience of the project and the recommendations derived from it – in the form of further adjustments or development of new functionalities – are collected.

The LTfLL project went through an important reorientation, which put different perspectives on the further development of the individual services. The development of the second (full) version of the services is replaced by an exploration of threads of the implemented functionalities. The first version showed that the different services produced software that was usable in real education contexts (directed by scenarios). The usefulness and relevance of the associated scenarios have been validated in pilot institutions. Questions however remain about the educational use in different contexts (transferability). During this process we came to a common understanding that the potential user group would be substantially if the project could offer “pick and mix” functionality based on Language Technology Tools. The ultimate goal became a “LEGO-lised” LTfLL tool kit of building blocks in combination with a visual programming environment to support end-user programming by educational practitioners.

The next section describes how the existing services can be modularised and widgetized. Subsequently, a first set of entities belonging to the LTfLL tool kit is specified and an outline sketched of a visual programming environment that could be designed to support the users in their threading approach.

### 7.2. Modularisation

To ease the communication with (potential) users and stakeholders the project team produced short (2 pages) descriptions for each service. These were based on the scenarios and they were used to check the match between the specifications and the working services (version 1). The combination of the six descriptions and the standardised template used helped to create overviews of the *functionality* provided by the main services, the possible *overlaps* as well as *complementary functions and data resources*.

A widget always combines user interface objects with the software functionality and it should act according to the requested behaviour of the third party site. As long as a widget is a real stand alone application the mutual dependencies can be limited. The use or not use of the widget can be controlled by switching it on or off. In these cases the main restrictions are on the available screen space left for the widget. During the widgetizing process it appeared that services, like for instance PolyCAFe and FLSS, offer parallel visualizations which conflict with the limited screen

spaces for a single widget approach. Developers were forced to take a next step of sub-widgetizing by modularisation of the functionality. However, this introduced more technical complexity, because it required a narrow cooperation between the different sub-widgets and a smooth data integration.

This challenge taken by the individual services (illustrative examples are WP6 and WP5.1) opens new opportunities for cooperation and data integration between (modules of) the services and it offers first lessons learnt for the different threading levels (Chapter 4). In addition, it enables the exchange of functionality and data resources between services. The price to be paid is that the subwidgets and threads make it rather complex for “any user” to assemble a coherent and usable interface to the (clustered) functionality. This may require stewards to create digital habitats (Wenger, White & Smith, 2009).

In the following table we present the results of an analysis from the service descriptions. The table is based on functional decomposition of the functionality as shown in the SUM diagrams. We have done this from a design perspective with a focus on the main stakeholders (teachers and learners). It is likely that after a detailed consideration of the technical implications these modules can be changed. Considering that in this phase we are exploring the threading approach with the design of a demonstrator and a conceptual validation, we can leave the intricacies of final modularisation, where the more technical issues, such as the possibility to isolate the functions, to the infrastructural WP2 and the developmental WP4/5/6.

In our table each distinct module will be described as widget or background service. Widgets have interface objects and will be noticed by the end users, background services are mainly for threading purposes. Background services are connected to one or more widgets and are important for the technical threading and the data integration. The numbering in the table indicates the origin of the sub-module: widget 6.2-3 is part of the original WP6.2 service. Sub-modules that conceptually share or overlap functionality are located in the same row. These are interesting candidates for reuse for the recombination of the individual services. From an infrastructural perspective WP2 should provide some common widgets like: single sign on [Widget 2-1], account management – role distribution [Widget 2-2].

	WP4.1	WP4.2	WP5.1	WP5.2	WP6.1	WP6.2
<b>Service</b>	Learner positioning	CONSPECT	PolyCAFe	PenSum	FFLSS-CES	iFLSS-LCP
<b>Sub Modules</b>	Phrase extractor/ scorer [bg service 4.1-1]	Concept matcher on the basis of concept maps [bg service 4.2-1]	Set the assignments concepts	Extract in texts main concepts <sup>3</sup> and main sentences [bg service 5.2-1]	Automatic document annotation (concepts & coreferences) - NLP annotation pipe [bg service 6.1-1]	Annotated document viewer [widget 6.2-1]
	Concept Annotation Pipe [bg service 4.1-1]	Conceptogram visualiser [widget 4.2-1]	Concept matcher (semantic similarity) [bg service 5.1-1]	Concepts/topics matcher on the basis of summaries [bg service 5.2-1]	Concept browser (map, tree, list) [widget 6.1-1]	Graph visualiser [widget 6.2-2]
	Concept matcher/ on the basis of curriculum annotation [bg service 4.1-1]		Concept extractor [widget 5.1-1]		Semi-automatic support for concept annotation [widget 6.1-4]	Tree visualiser [widget 6.2-3]

<sup>3</sup> In other documents notions/topics called.

	WP4.1	WP4.2	WP5.1	WP5.2	WP6.1	WP6.2
	Level measurement by speech genre recogniser (phrase matcher) [widget 4.1-2]	Evidence collector [bg service 4.2-1]	Chat threads collector [bg service 5.1-1]	Locate missing and irrelevant concepts / sentences in learner text [bg service 5.2-1]	Semi-automatic support for discourse annotation	Crawler for social networks based on user settings [widget 6.2-4]
	List of concepts (Common, Missing, Additional) [widget 4.1-1]	Evidence preprocessor [bg service 4.2-1]	NLP pipe preprocessor [bg service 5.1-1]	Check coherence of learner text		Format conversion [bg service 6.2-2]
	Ontology & lexicon enrichment service [bg service 4.1-2]	Compare or combine conceptual models [widget 4.2-2]	Discourse analyser [widget 5.1-2]	Notebook for learning questions [bg service 5.2-1]		Ontology enrichment (incl Concept disambiguator) [bg service 6.2-1]
	Semi-automatic corpus building [bg service 4.1-1]	Identify shortcomings and misconceptions [bg service 4.2-1]	Concepts & phrases inspector [widget 5.1-3]	Match analysis & self assessment [widget 5.2-2]	Textual search engine	Textual search engine
	Suggest reading materials (for gaps) [widget 4.1-3]	Suggest reading materials	Chat discussion visualiser [widget 5.1-4]	Suggest additional reading materials [widget 5.2-2]	Document ranking	Resource ranking [bg service 6.2-1-2]
	Positioning on required competences [widget 4.1-4]	Positioning in own group (find outliers) [widget 4.2-3]	Forum discussion visualiser [widget 5.1-5]		Semantic search engine (incl. query expansion) [widget 6.1-2]	Semantic search engine (incl. query expansion & visualisation format) [widget 6.2-5]
			Manual feedback tuner		Phrase-concept linker [widget 6.1-5]	Concept-tags linker [bg service 6.2-1]
			Social network analyzer [bg service 5.1-1]		Annotation editor [widget 6.1-4]	Generate user profile [widget 6.2-6]
			Readability rater [widget 5.1-6]			Definition Viewer [widget 6.2-7]
						List of ranked and tagged peers [widget 6.2-8]
	List individual learning path [widget 4.1-5]	Feedback system	Feedback and grading system	Feedback to learner Texts [widget 5.2-3]	List of (ranked) documents [widget 6.1-2]	List of ranked and annotated documents (content/peers) [widget 6.2-9]
<b>General Corpus</b>	English, German, Bulgarian (v2)	English, Dutch (v2)	English, Romanian (v2)	French, English (v2)	English, Bulgarian (v2)	English, Dutch (v2)
<b>Data</b>	Background Corpus, Reference corpus IT Job, course and question data Distincted phrases Expert utterances, public utterances Domain ontology IT LSA Vector Space IT Lexicon IT Predefined position requirements Classified answers WUW LSA Vector	Background Corpus Reference corpus Medicins Key Concepts (subset of ontology?) WUW LSA Vector Space Medicin MeSH codes Concept clusters Individual conceptual models Emerging group models Reference model	Background Corpus Domain ontology HCI, WUW LSA Vector Space HCI Chat logs Linguistic ontology (WordNet) Discussion/forum posts Selected assignment concepts Feedback utterances Grading rules	Background Corpus Course texts DB of syntheses & learning questions Cognitive computational models WUW LSA Vector	Domain ontology IT Repository Learning materials Concept grammars Lexicons	Domain ontology Computing Lexicons RDF repository Structured information of DBpedia Crawled triples (Delicious) Ontologies related to social networks and/or semantic web (SIOC, FOAF, SKOS, SCOT, MOAT) Various similarity measures Members of social networks

### ***7.3. Modularisation and recombination***

The functional decomposition in modules, widgets and background services as outlined in 7.2 defines the starting point for the recombination efforts. Some of these can be easily combined into a single module used for different services, others will need additional development resources to make them more generic, or to enable more flexibility for the input and output data by changing their APIs. In the DoW a choice was made to limit, as much as possible, the mutual dependencies between the development teams. This was done to allow each service to explore and find its own development path and to manage the complexity.

The limited development resources left, force the LTfLL project to focus and to prioritise all its development efforts. Parts of these are needed to develop collaboratively a demonstrator of the thread. Additional efforts are still needed for the individual services to enable the next validation (see D7.3) round with larger user groups during prolonged time frames. There is no room left for extra modularisation efforts. This will therefore become part of the roadmap or research calendar. As project we will strive to collect more information to support the prioritisation. In the first round we found interesting data about the use. We got insights in what functionality the users liked and what attracted or surprised them. The next round will enlarge these insights and inform us more about the performance, stability etc, under realistic usage conditions.

The selected language technologies and tools put requirements on the data used for the input and output. Whenever these are annoying for users and endanger the usability (e.g. PenSum requires unformatted texts), these should be removed with preprocessing (converters). Our leading approach of threading makes this converting even more important. A generic solution should be taken to prevent us from continuously changing the modules to meet new data formats. Specific converter widgets should be designed and developed to enable data integration and to guarantee that the API of each module will be fed with the correct data.

We expect that in the beginning threading will be done by developers, who are able to connect the different widgets and to enable common access to data and data transfer. However if we want to exploit our LTfLL services and tool kit on a larger scale, we should offer end user programming with additional support. In the next section we sketch our first ideas of a possible visual programming environment.

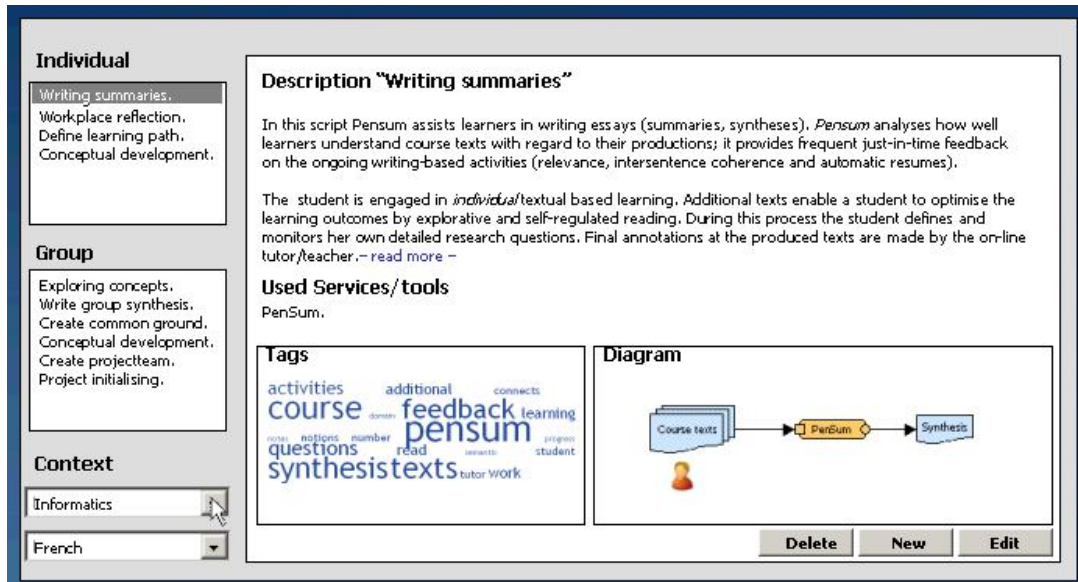
### ***7.4. Visual programming environment for recombination***

By the introduction of the visual programming environment we will support the possibilities to:

- reuse and to adapt existing threads
- create new threads based by configuring individual services
- create new threads and services from scratch

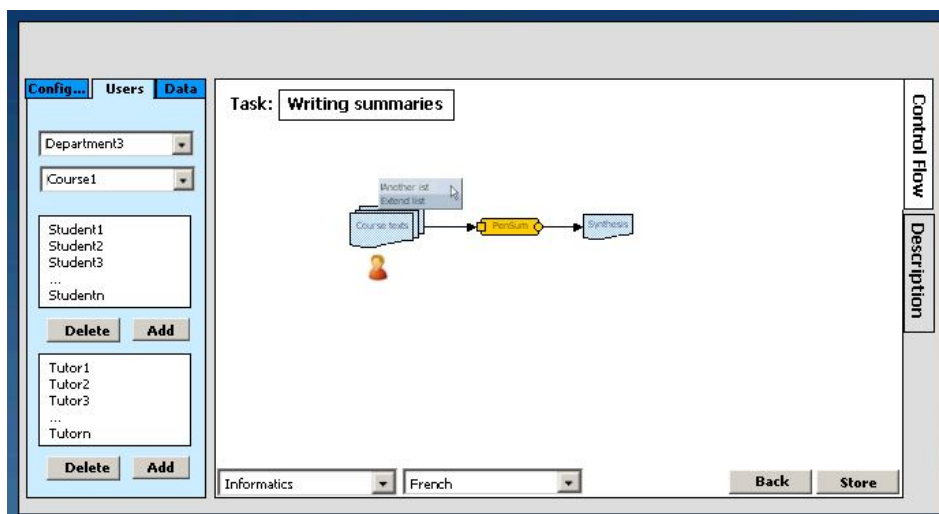
The programming environment combines a drawing canvas for the control flow with tabbed tool palettes to configure or to link and position the entities (e.g. data, converters and modules). The main entity of the environment is the learner task. The task is a specific combination of domain, control flow and set of functionalities. By changing one of these three elements a new learner task will be created. Each learner task has a description based on text and tags. At the highest level of the environment a choice has to be made to configure existing learner tasks or to edit

these. In the following picture we will show a possible list of existing learner tasks to be edited or configured.



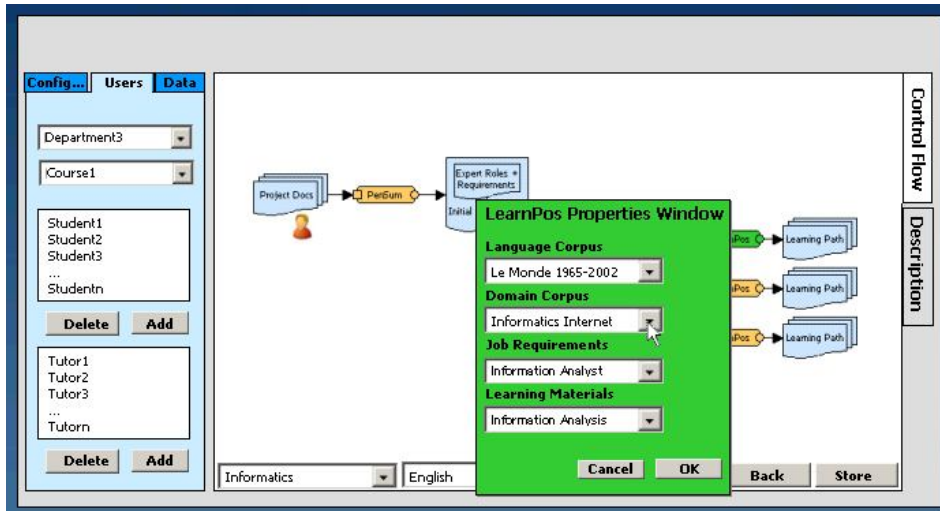
The screenshot shows a configuration window for a task named "Writing summaries". On the left, there are three sections: "Individual" (with sub-section "Writing summaries" containing "Workplace reflection", "Define learning path", and "Conceptual development"), "Group" (with "Exploring concepts", "Write group synthesis", "Create common ground", "Conceptual development", "Create project team", and "Project initialising"), and "Context" (with dropdowns for "Informatics" and "French"). The main area contains a "Description" of the task, "Used Services/tools" (listing "PenSum"), and a "Diagram" showing a flow from "Course texts" to "PenSum" to "Synthesis". A "Tags" section contains a word cloud with terms like "course", "feedback", "learning", "questions", "synthesis", "texts", "read", "tutor", "work", "number", "connects", "additional", "activities", "course", "pensum", "read", "tutor", "work", "number", "connects", "additional", "activities", "course", "pensum", "read", "tutor", "work", "number", "connects", "additional", "activities", "course", "pensum", "read", "tutor", "work". At the bottom right are "Delete", "New", and "Edit" buttons.

If the user selects to configure the writing summaries task (using PenSum as service) only the tool palette will be offered to configure e.g. to manage the learning groups and tutors and to select the language or domains.

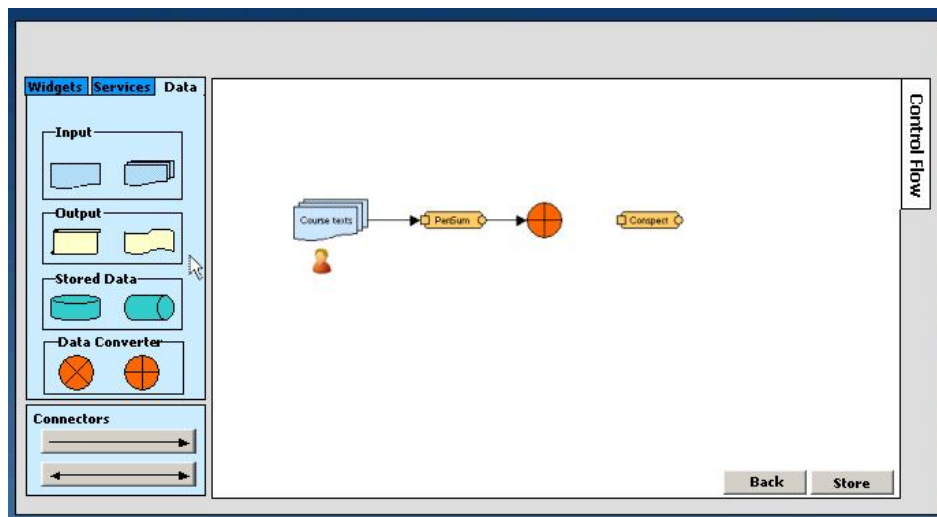


The screenshot shows a configuration window for a task named "Writing summaries". On the left, there are two sections: "Users" (with "Department3" and "Course1" dropdowns, and a list of "Student1", "Student2", "Student3", "...", "Studentn" with "Delete" and "Add" buttons) and "Tutors" (with a list of "Tutor1", "Tutor2", "Tutor3", "...", "Tutorn" with "Delete" and "Add" buttons). The main area contains a "Control Flow" diagram showing a flow from "Course texts" to "PenSum" to "Synthesis". The diagram also includes a "Another list" and "Extend list" button. At the bottom are "Informatics" and "French" dropdowns, and "Back" and "Store" buttons.

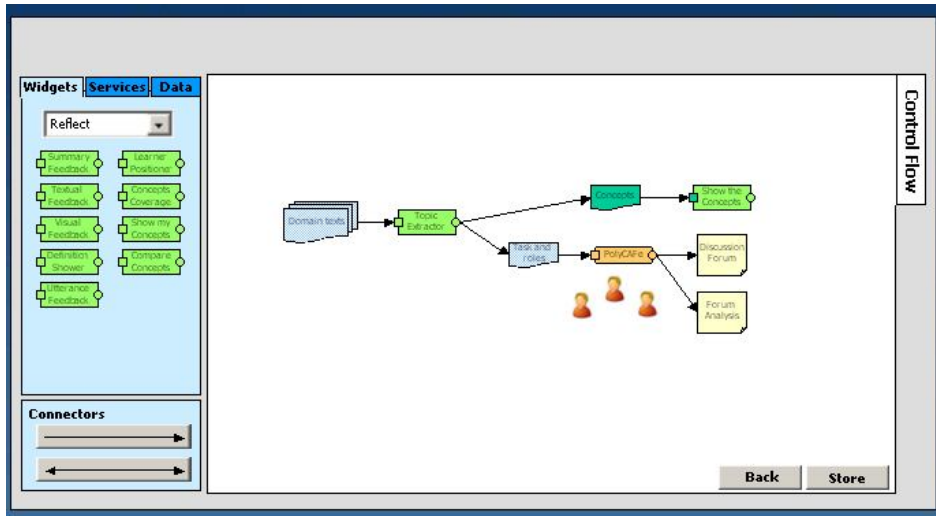
For the threading examples where different services are lined up in one control flow the different services can require additional configuration settings. This can be done using popup menus as shown in the following picture, where LearnPos is configured for positioning.



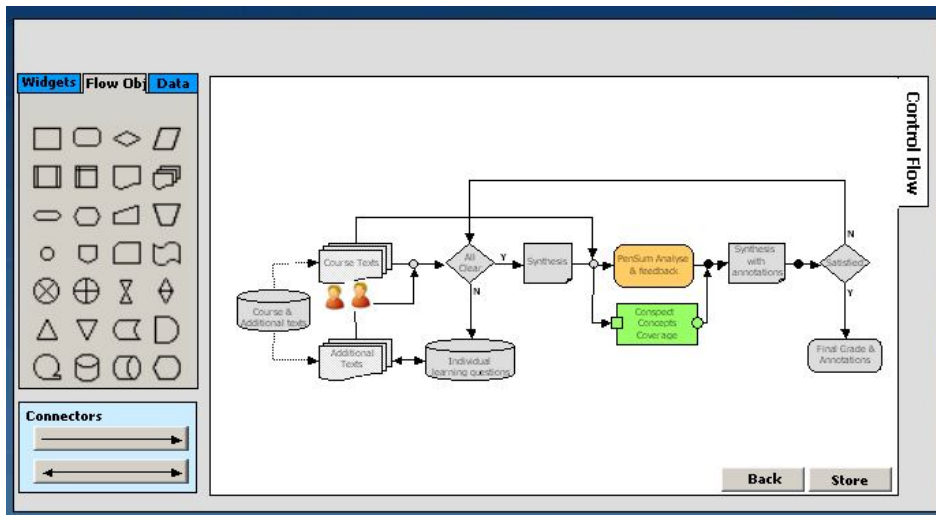
Whenever the user wants to create new threads the tool palette shows all possible entities: services, widgets, data and the connectors. The programming environment uses its knowledge of the input and output requirements to allow connections. If the output fails to match the requirements of the input a data converter widget can be used to do that conversion. Only converters that are able to work with the PenSum and CONSPECT data will be listed. The following picture shows how such a converter supports the use of PenSum results as input to CONSPECT.



In the examples given of the visual programming environment we have seen how the individual services can be placed in one control flow. The modularisation of the services into subwidgets enables a next step of Pick and Mix and allows the combination of services and subwidgets in the same control flow to develop new threads (see next picture).



To conclude the outline of the visual programming we want to show a last example. This shows the real recombination, because it shows how one of the original services, namely PenSum, is rebuilt using the LTfLL toolkit. With this recombination redundant functionality can be left out or extra functionality (offered by one of the other services) can be put in. Rebuilding of the service could be done as realistic verification task for modularization. Such successful rebuilding will proof the concepts of modularisation and recombination convincingly. In the next picture we show the rebuilding process.



The given examples give an impression of the concepts behind the visual programming environment. However, for the LTfLL project this environment will remain conceptual. Extend (Rivera, 1997) uses the same approach of offering a visual programming environment to support interactive modelling. In our case especially the openness of the environment will be important. It will be a challenge to allow the import of other third party widgets to extend the functionality even further.

## 8. References

- Alpay, L., Giboin, A., & Dieng, R. (1998). Accidentology: an example of problem solving by multiple agents with multiple representations. In M.W. Van Someren, P. Reimann, H.P.A. Boshuizen, & T. de Jong (Eds.), *Learning with multiple representations* (pp. 152-174). Amsterdam: Pergamon.
- Baker, M. (2003). Computer-mediated argumentative interactions for the co-elaboration of scientific notions. In: Andriessen, J., Baker, M., Suthers, D. (Eds.), *Confronting cognitions: Arguing to learn*. Kluwer Academic Publishers, Dordrecht, The Netherlands. pp. 47-78.
- Barkley, E.F., Cross, K.P., & Major, C.H. (2005). *Collaborative learning techniques: A handbook for college faculty*. San Francisco: Jossey-Bass.
- Beers, P.J., Boshuizen, H.P.A., Kirschner, P.A., & Gijsselaers, W.H. (2005). Computer support for knowledge construction in collaborative learning environments. *Computers in Human Behavior*, 21, 623-643.
- Bosworth, K. (1994). Developing collaborative skills in college students. In K. Bosworth & S.J. Hamilton (Eds), *Collaborative learning: Underlying processes and effective techniques*. New Directions in Teaching and Learning. No. 59. San Francisco: Jossey-Bass.
- Burstein, J., Kaplan, R., Wolff, S., & Lu, C. (1996). Using lexical semantic techniques to classify free-responses, *Proceedings from the SIGLEX19666 workshop, Annual meeting of the Association of Computational Linguistics*, University of California, Santa Cruz.
- Chi, M.T.H., Feltovich, P.J., & Glaser, R. (1981). Categorization and representation of physics problems by experts and novices. *Cognitive Science*, 5, 121-152.
- Clewley, G. and Bowen-Clewley, L. (2005). *A report on multidisciplinary approaches in public health*, Competency International Ltd.
- Conklin, E.J. (2002). *Making sense of fragmentary information: Compendium and the intelligence community*. Edgewater, MA: CogNexus Institute.
- Conklin, E.J., & Weil, W. (1997) *Wicked problems: naming the pain in organizations*. <http://www.gdss.com/wp/wicked.htm>. Accessed 05-06-2001.
- Crossfunctional teams*: <http://best.berkeley.edu/~pps/pps/teams.html> (accessed 18-5-2010).
- De Bakker, G., Van Bruggen, J., Sloep, P., Jochems, W. (in press). Introducing the SAPS model and a corresponding allocation mechanism for synchronous online reciprocal peer support activities. *Journal of Artificial Societies and Social Simulation*.
- Dillenbourg, P. (1996). Distributing cognition over humans and machines. In S. Vosniadou, E. de Corte, R. Glaser, & H. Mandl (Eds.), *International perspectives on the design of technology-supported learning environments* (pp. 165-184). Mahwah, NJ: Lawrence Erlbaum Associates Publishers.
- Dillenbourg, P. (2002). Over-scripting CSCL: The risks of blending collaborative learning with instructional design. In *Three worlds of CSCL: Can we support CSCL* (pp. 61-91). Heerlen, The Netherlands: Open University of the Netherlands.
- Dong, A. (2004). Quantifying coherent thinking in design: a computational linguistics approach. In J. Gero (Ed.), *Design Computing and Cognition '04* Dordrecht, The Netherlands: Kluwer Academic Publishers.

- Dong, A. (2005). The latent semantic approach to studying design team communication. *Design Studies*, 26, 445-461.
- Duffy, T.M., Dueber, B., & Hawley, C. (1998). *Critical thinking in a distributed environment: a pedagogical base for the design of conferencing systems* (CRLT Technical report No 5-98). Bloomington. In: Indiana University, Center for Research on Learning and Technology.
- Foltz, P.W., Gilliam, S., & Kendall, S. (2000). Supporting content-based feedback in online writing evaluation with LSA. *Interactive Learning Environments*, 8, 111-129.
- Foltz, P.W., Laham, D., & Landauer, T. (1999). Automated essay scoring: Applications to educational technology. *Proceedings of EdMedia '99, Seattle*.
- Garrison, D.R., Anderson T., & Archer W. (2000) Critical inquiry in a text-based environment: computer conferencing in higher education. *The Internet and Higher Education*, 2, 87-105.
- Johnson, D.W., Johnson, R.T., & Smith, K.A. (1991). *Cooperative learning: Increasing college faculty instructional productivity*. ASHE-ERIC Higher Education Reports. No. 4. Washington, DC: George Washington University.
- Kintsch, E., Steinhart, D., Stahl, G., LSA research group, Matthews, C., & Lamb, R. (2000). Developing summarization skills through the use of LSA-based feedback 30. *Interactive Learning Environments*, 8, 87-109.
- Nivelstein, F., Van Gog, T., Boshuizen, H.P.A., & Prins, F.J. (2008). Expertise-related differences in ontological and conceptual knowledge in the legal domain. *European Journal of Cognitive Psychology*, 20, 1043-1064.
- Pfeiffer, S.I. (1981). The problems facing multidisciplinary teams: As perceived by team members. In *Psychology in the Schools*, 18, 330-333.
- Rittel, H.W. J., & Webber, M.M. (1984). Planning problems are wicked problems. In N. Cross (Ed.), *Developments in design methodology* (pp. 135-144). Chichester: John Wiley & Sons. (published earlier as part of 'Dilemmas in a general theory of planning', *Policy Sciences*, 4, 1973, (155-169).
- Rivera, J. (1997.) Modeling With Extend™. In. S. Andradóttir, K.J. Healy, D.H. Withers, and B.L. Nelson (Eds), *Proceedings of the 1997 Winter Simulation Conference*. IEEE, Piscataway, NJ.
- Rosson, M., Carroll, J. (2002). Usability engineering: Scenario-based development of human-computer interaction, San Francisco: Morgan Kaufmann.
- Selvin, A.M. (2003). Fostering collective intelligence: helping groups use visualized argumentation. In P.A. Kirschner, S.J. Buckingham Shum, & C.S. Carr (Eds.), *Visualizing Argumentation: Software tools for collaborative and educational sense-making* (pp. 137-163). Springer-Verlag.
- Simon, H.A. (1996) *The sciences of the artificial* (3rd Ed). Boston: MIT Press.
- Sloffer, S.J., Dueber, B., & Duffy, T.M. (1999). *Using asynchronous conferencing to promote critical thinking: two implementations in higher education* (CRLT Technical Report No. 8-99). Bloomington. In: Indiana University, Center for Research on Learning and Technology.
- Stahl, G. (2006), *Group Cognition: Computer support for building collaborative knowledge*. Cambridge, MIT Press.
- Teachout, M., Segó, D., & Ford, J. (1997). An integrated approach to summative evaluation for facilitating training course improvement. *Training Research Journal*, 3, 169-184.
- Van Bruggen, J.M., Boshuizen, H.P.A., & Kirschner, P.A. (2003). A cognitive framework for cooperative problem solving with argument visualization. In P.A. Kirschner, S.J.

- Buckingham Shum, & C.S. Carr (Eds.), *Visualizing argumentation: Software tools for collaborative and educational sense-making* (pp. 25-47). London: Springer.
- Van Rosmalen, P., Sloep, P., Brouns, F., Kester, L., Kone, M., and Koper, R. (2006). Knowledge matchmaking in Learning Networks. *British Journal of Educational Technology*, 37, 881-895.
- Van Rosmalen, P., Sloep, P., Kester, L., Brouns, F., De Croock, M., Pannekeet, K., & Koper, R. (2008). A learner support model based on peer tutor selection. *Journal of Computer Assisted Learning*, 24, 74-86.
- Wan, D., & Johnson, P. M. (1994). Computer supported collaborative learning using CLARE: the approach and experimental findings. *CSCW '94. Proceedings of the conference on Computer supported cooperative work* (pp. 187-198).
- Wenger, E., White, N., & Smith, J.D. (2009) *Digital Habitats: stewarding technology for communities*. Portland, OR: Cpsquare.
- Westera, W., & Sloep, P.B. (1998) The Virtual Company: Toward a self-directed, competence-based learning environment in distance education. *Educational Technology* 38, 1, 32-37.