



Objectiver 1.5.0

A Technical Overview

Information for Analysts, Method Engineers and Consultants

A POWER TOOL TO ENGINEER YOUR BUSINESS & TECHNICAL REQUIREMENTS

Based on nearly 10 years of practical experience and development in the Requirements Engineering field, Cediti is releasing a new generation tool, Objectiver, which is dedicated to requirements engineers and more generally to all those having to achieve high-level, goal-oriented analyses.

Objectiver lets you identify and **model** your requirements. It will support you with the communication and validation of your model. Then, it can generate a complete, structured and easy-to-read requirements document.

After a short introduction to Kaos, this document presents the main features which make Objectiver a unique tool.

REQUIREMENTS ENGINEERING AND THE KAOS METHOD

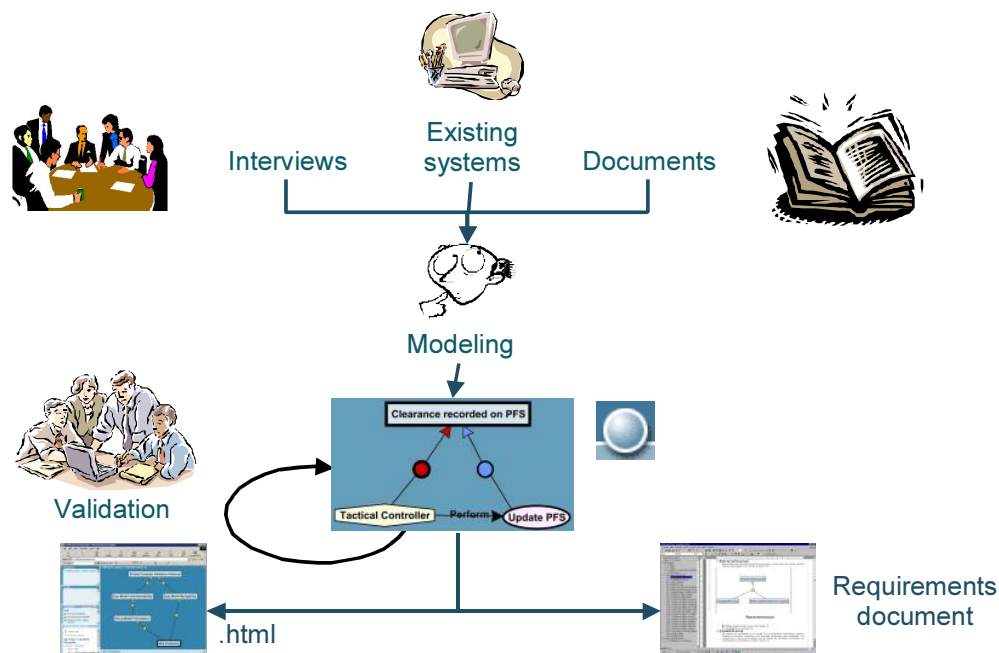
Most of the software engineering methodologies currently in use are dedicated to the specification of a solution and do not really address the description of the problem, that is, specifying the goals to be met, describing the application domain, deciding who will perform which actions to satisfy the goals, etc.; in one word, to perform a requirement analysis.

Issued from the scientific community, Kaos is one of the most prominent Requirement Engineering methods. A requirements analysis in Kaos is broken down into three phases:

1. gathering of the information to be used as a guide for the goals to be achieved
2. modeling
3. drawing up of a report (specifications, synthesis, IT master plan, etc.)

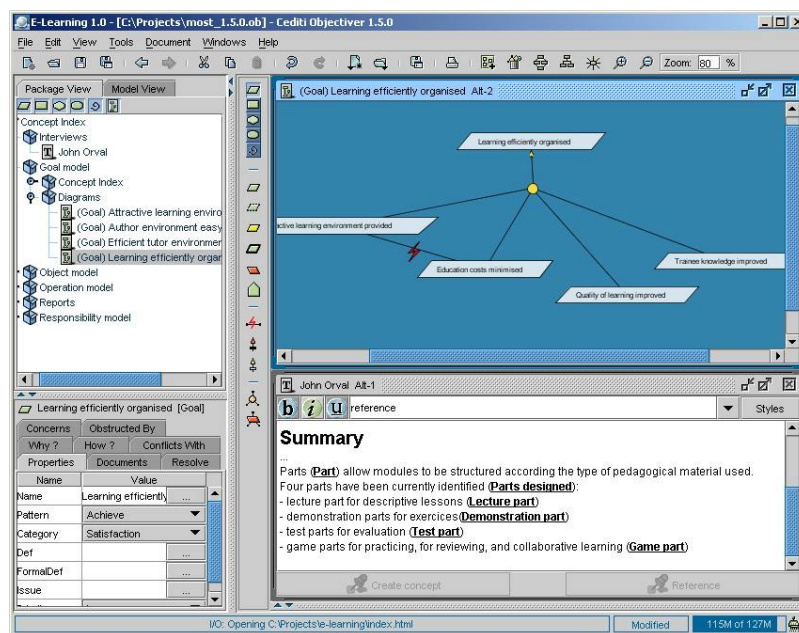
Today, Objectiver is the only tool which fully supports the Kaos method, its graphical notations and its analysis process.

The following figure outlines the KAOS analysis process starting from the information gathering to the generation of a requirements document.



OBJECTIVER OVERVIEW

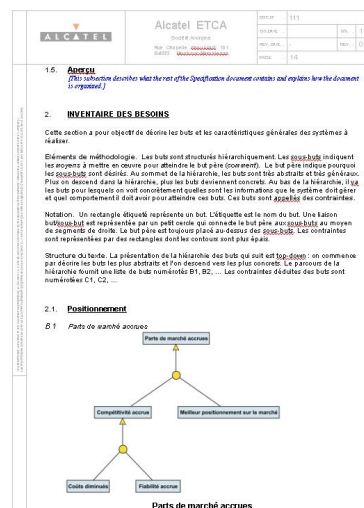
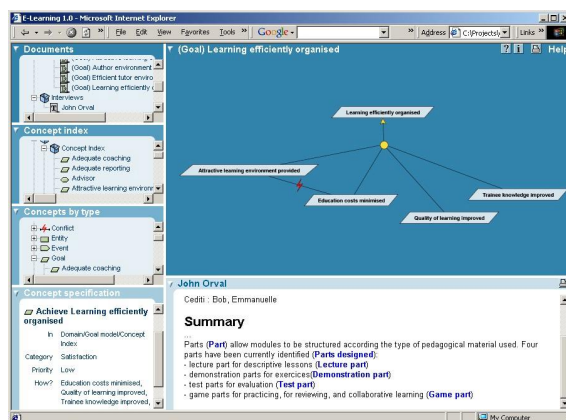
The figure below shows a snapshot of the tool and two outputs generated from it.



The Objectiver tool lets you acquire and model your requirements.

Requirements models can be published as a stand-alone web representation, and displayed in an Internet browser.

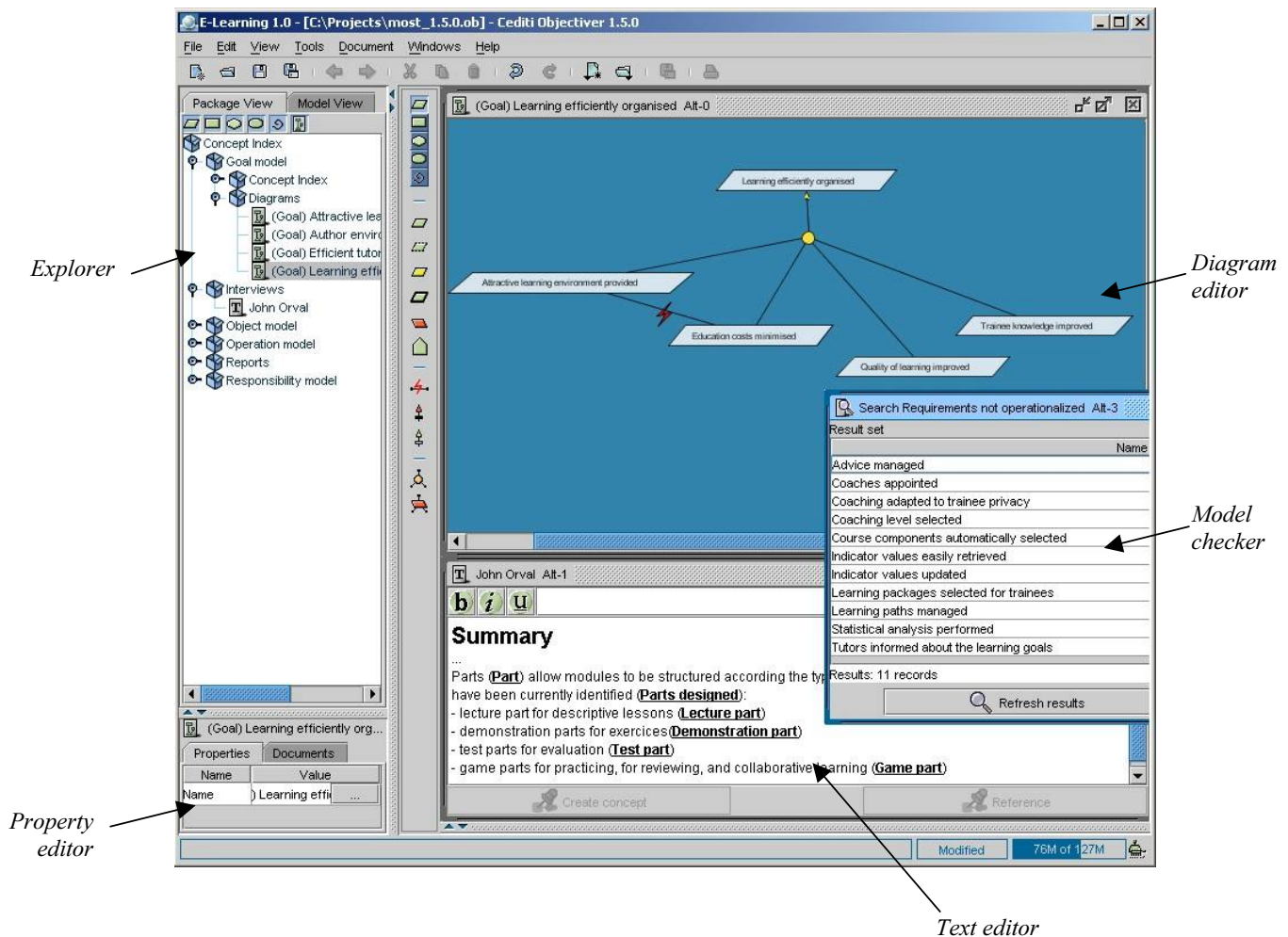
Requirements documents are generated from the model (PDF or RTF format).



BUILDING THE MODEL

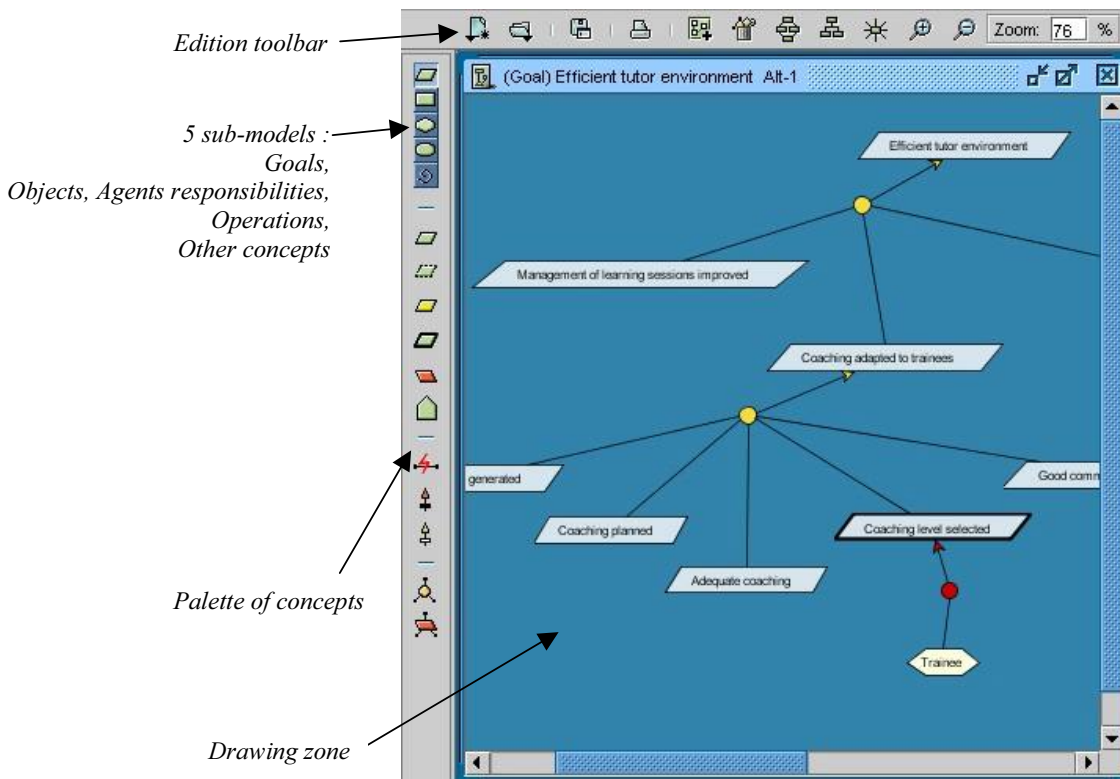
The Objectiver application provides analysts with a complete set of tools to build a requirements model:

- a [diagram editor](#) to draw the concepts and their relationships
- a [text editor](#) to record interview summaries and to associate descriptive texts to diagrams
- a [property editor](#) to set value of predefined properties and add user-defined attributes
- an [explorer](#) to list, sort and filter concepts and documents
- a [model checker](#) to check the completeness and quality of the model



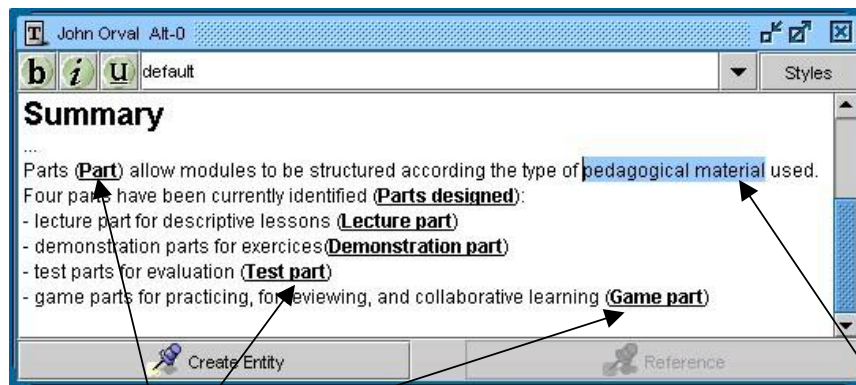
THE DIAGRAM EDITOR

Diagrams are an expressive way to represent concepts and their relationships. The palette offers to the analyst all concepts types defined by the Kaos method.



THE TEXT EDITOR

A KAOS model contains texts such as interview summaries or diagrams descriptions. The text editor can be used to format a text, and add references to model concepts. A concept reference is a simple, yet powerful way to put in place traceability from interviews to the final requirements document.

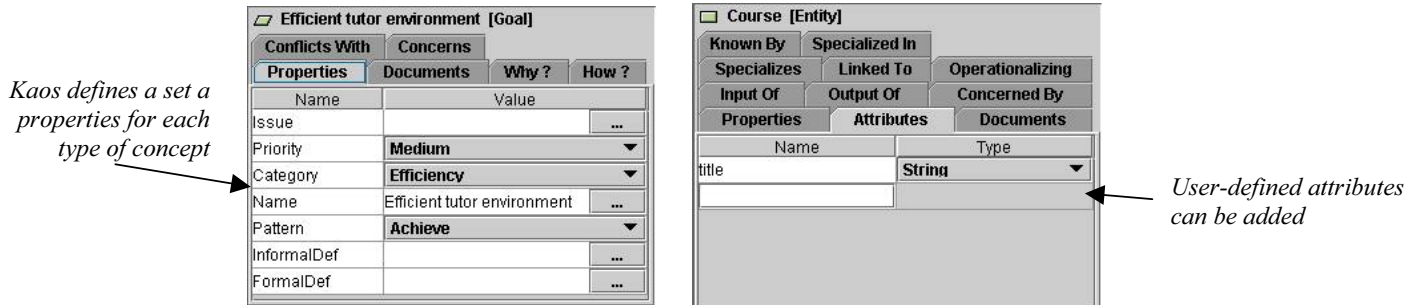


A reference to a known concept can be added to a text by drag&drop or auto-completion

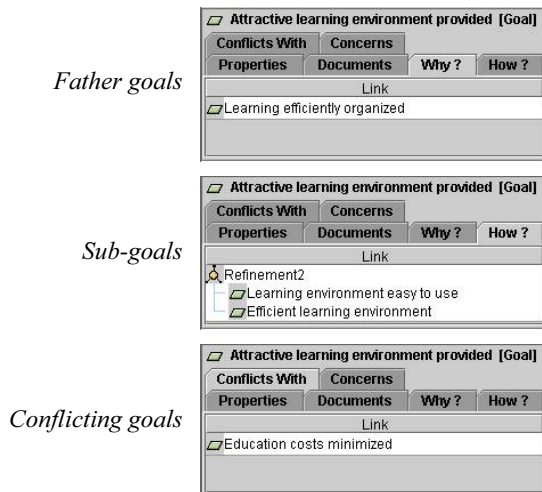
A concept can be created and its reference added in a single step.

THE PROPERTY EDITOR

The property editor is the place where the properties of concepts are displayed and edited.



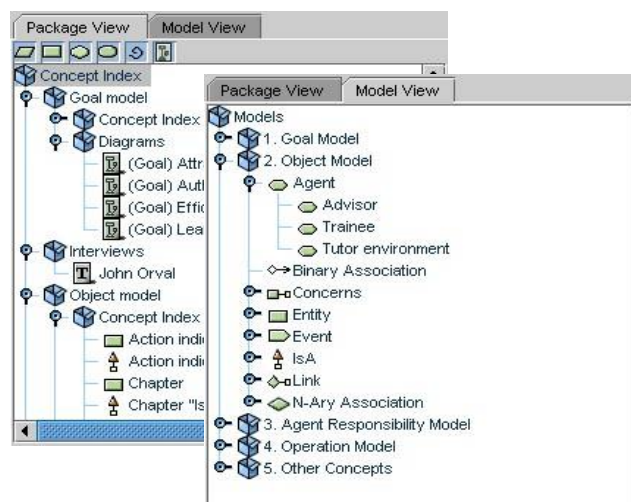
The property editor is also an easy way to check and navigate relationships between concepts in the model. Each tab shows a list a related concepts. In example :



THE EXPLORER

The explorer provides two hierarchical views on a KAOS model. The first one allows you to structure the model into packages. The second one shows a classification of concepts according to the KAOS method.

Its content can be sorted, filtered and searched in a single click.

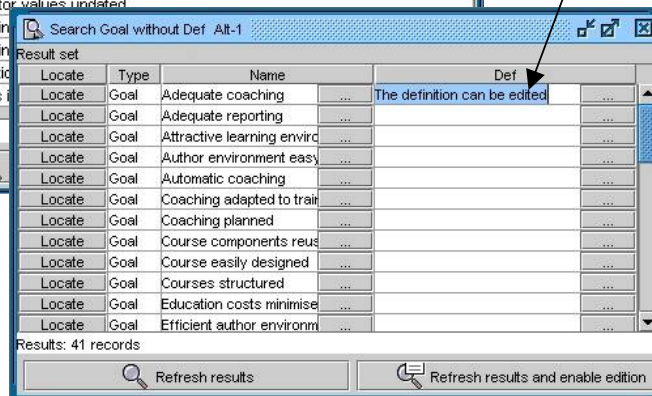
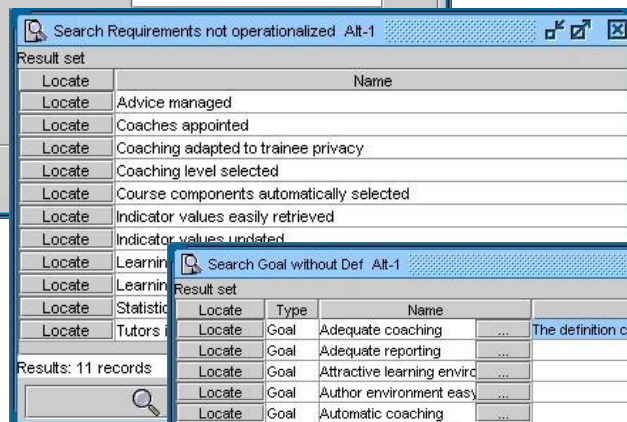
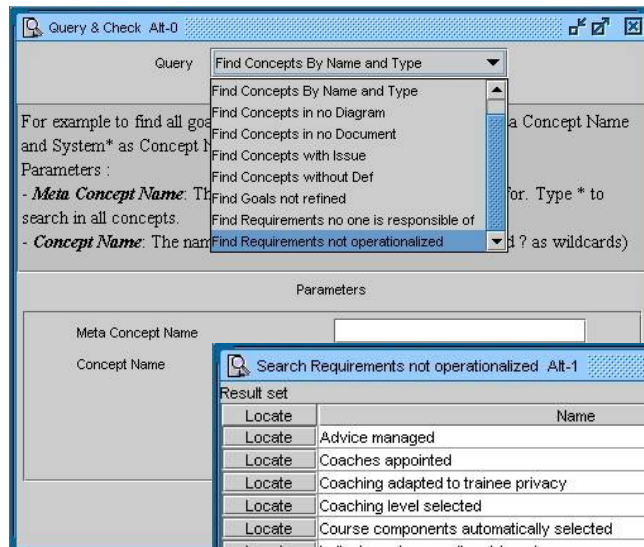


THE MODEL CHECKER

The completeness and quality of your model can be checked.

At any moment, Objectiver can show you :

- goals which are not refined
- requirements under the responsibility of no agent
- non-operationalized requirements
- concepts never appearing in a document
- undefined concepts
- pending issues



Directly edit the check result
to easily complete your model

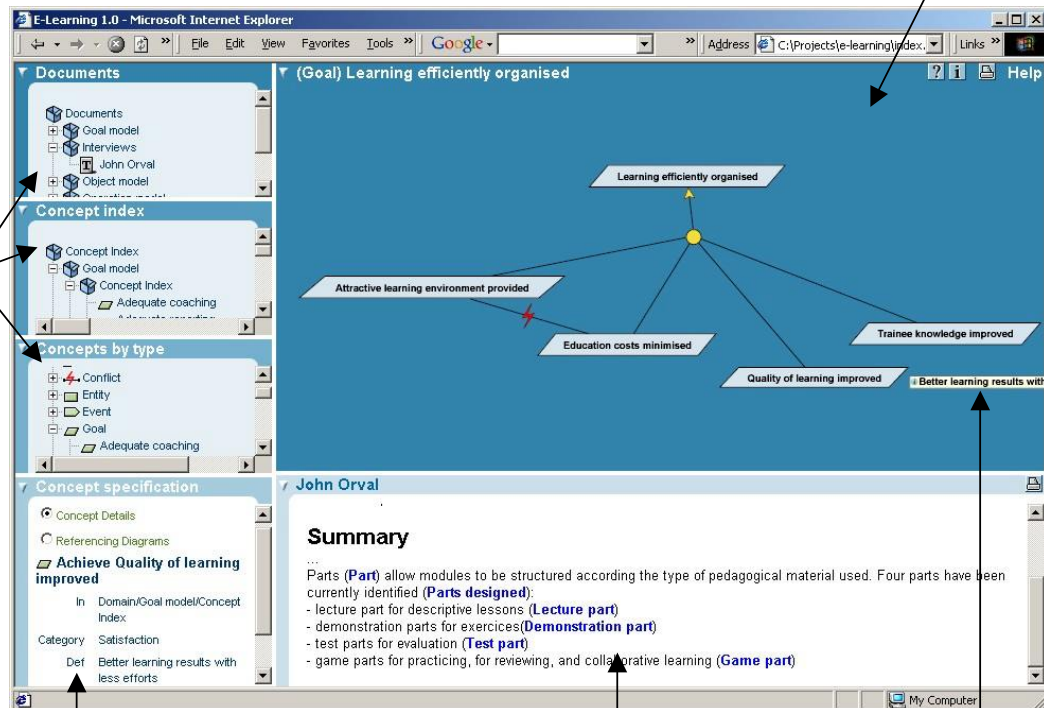
PUBLISHING ON THE WEB

Objectiver can publish a model as a stand-alone interactive web representation. It is composed of hyperlinked HTML and SVG pages, and is viewed in a Internet browser*.

You can generate it at any time, and distribute it freely. The web representation has been proved by experience as a very efficient support for communication and for validation with the stakeholders.

All model concepts in indexes, property, text, and diagram viewers are hyperlinked, and automatically located in diagrams. Diagrams can be zoomed, panned and printed easily; they do react dynamically to show important concept properties.

Interactive diagram viewer



Indexes

Text viewer with hyperlinks to concepts

Pop-ups show important properties

shows details of each concept, links to related concepts, and referencing documents

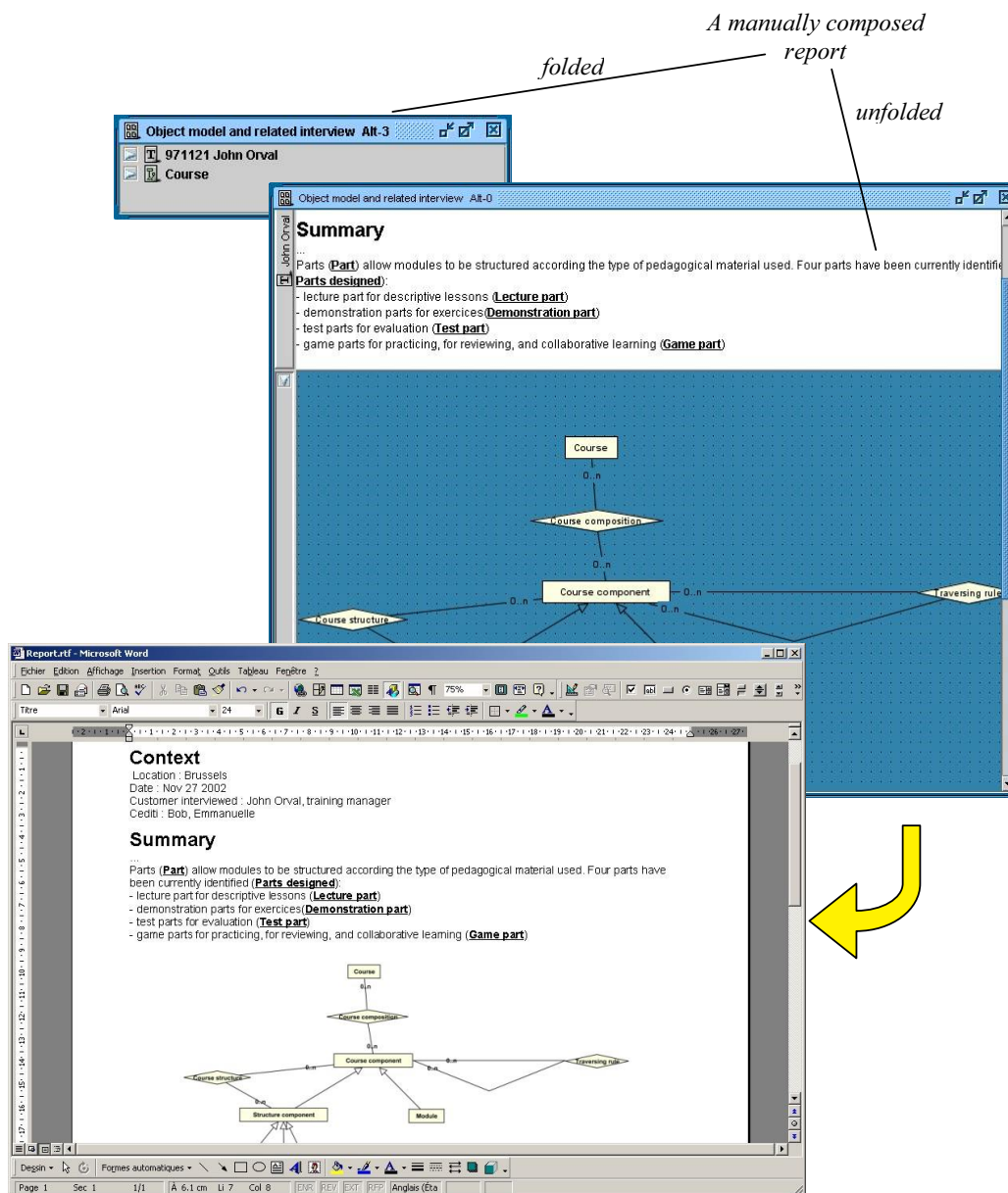
* Requires SVG Viewer 3.0 plug-in freely available from Adobe and Microsoft Internet Explorer 5.5 or higher.

GENERATING DOCUMENTS

Objectiver can generate different kinds of reports and export them in multiple document formats such as PDF and RTF.

A report is an Objectiver document that contains a sequence of other Objectiver documents such as diagrams and texts. Reports can be manually or automatically composed, then automatically generated in PDF or RTF. Typical reports are :

- glossaries, and lists of issues
- an explanation of a diagram, a package or multiple packages
- a requirements document following a standard template (IEEE 830)
- a requirements document following a template customized to company standards



A complete Requirements Document can be generated following a standard template. An assistant will guide you in the process, and will help you defining the structure of the document.

IEEE Computer Society Std 830-1998

IEEE Recommended Practice for Software Requirements Specification

1. Introduction

1.1. Document purpose

1.2. System purpose

1.3. Definitions, acronyms and abbreviations

1.4. References

1.5. Overview

2. Overall Description

2.1. System perspective

2.2. User requirements

2.3. User characteristics

2.4. Constraints

2.5. Assumptions and Dependencies

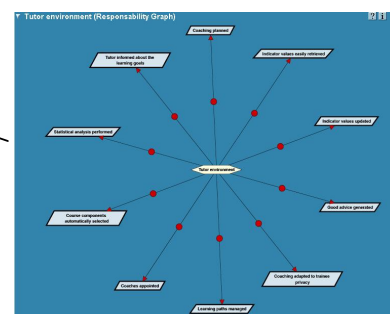
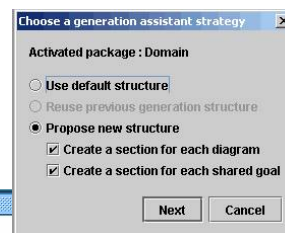
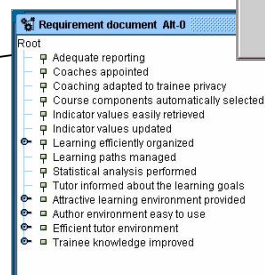
2.6. Apportioning of requirements

3. System Requirements

*A glossary of all
concepts is
generated*



*The assistant proposes
the order of goals description,
and generates missing goals diagrams*



In a single additional operation, publish the whole Requirements Document to a complete, easy to read RTF or PDF document.

The exported document features

- formatted text and styles
- customizable document template
- cross-references and hyperlinks
- table of contents and indexes
- scalar or rasterized diagrams

	Alcate ETCA Société R. Anonyme Rue Charrière 0000000000, Tel. 8-8000 00000000000000000000	DOCUMENT : III ETS DATE : REV. DATE : REVISION : 1/0	PSE : 1 REV. : 0
---	---	---	---------------------

1.5. **Aperçu**
[This subsection describes what the rest of the Specification document contains and explains how the document is organized.]

2. INVENTAIRE DES BESOINS

Cette section a pour objectif de décrire les buts et les caractéristiques générales des systèmes à réaliser.

Éléments de méthodologie. Les buts sont structurés hiérarchiquement. Les sous-buts indiquent les moyens à mettre en œuvre pour atteindre le but père (comment). Le but père indique pourquoi les sous-buts sont désirés. Au sommet de la hiérarchie, les buts sont très abstraits et très généraux. Plus on descend dans la hiérarchie, plus les buts deviennent concrets. Au bas de la hiérarchie, il y a les buts pour lesquels on voit concrètement quelles sont les informations que le système doit gérer et quel comportement il doit avoir pour atteindre ces buts. Ces buts sont appelés des contraintes.

Notation. Un rectangle étiqueté représente un but. L'étiquette est le nom du but. Une liaison butsousbut est représentée par un petit cercle qui connecte le but père aux sous-but au moyen de segments de droite. Le but père est toujours placé au-dessus des sous-buts. Les contraintes sont représentées par des rectangles dont les contours sont plus épais.

Structure du texte. La présentation de la hiérarchie des buts qui suit est top-down : on commence par décrire les buts les plus abstraits et l'on descend vers les plus concrets. Le parcours de la hiérarchie fournit une liste de buts numérotés B1, B2, ... Les contraintes déduites des buts sont numérotées C1, C2, ...

2.1. Positionnement

B1 Parts de marché accrues

```

graph TD
    A[Parts de marché accrues] --> B[Compétitivité accrue]
    A --> C[Meilleur positionnement sur le marché]
    B --> D[Coûts diminués]
    B --> E[Fiableté accrue]
  
```

Parts de marché accrues