Fraga, A., Llorens, J. & Génova, G. Towards a Methodology for Knowledge Reuse Based on Semantic Repositories. Inf Syst Front 21, 5–25 (2019).

DOI: https://doi.org/10.1007/s10796-018-9862-7

INFORMATION SYSTEMS FRONTIERS. 21:5-25. 2019

# Towards a Methodology for Knowledge Reuse Based on Semantic Repositories

Anabel Fraga, Juan Llorens, Gonzalo Génova

Universidad Carlos III de Madrid, Departamento de Informática,
Av. Universidad 30, Leganés,
28911 Madrid, Spain
`{afraga, llorens, ggenova}@inf.uc3m.es`
`http://www.kr.inf.uc3m.es`

**Abstract.** Although reuse is generally considered a good practice within software engineering, several problems dissuade its industrial application and a new viewpoint is needed. This paper presents a new perspective of reuse based on improved retrieval techniques for semantic content (knowledge). This approach, called Universal Knowledge Reuse Methodology (UKRM), drops the investment costs needed in systematic reuse, including the cost of traceability in the process, and reduces the chaos of ad-hoc reuse. UKRM makes reuse independent of the type of content, the context where it will be reused, and even the user that demands it. The paper includes an incremental experiment in order to validate the feasibility of this proposal.

**Keywords:** reuse, systematic reuse, ad-hoc reuse, knowledge reuse, reuse methodology, semantic repositories, ROI problems, Reuse and integration in domain transference.

## 1 Introduction

Currently, systematic software reuse is not interesting, profitable or easy for practitioners, and a new viewpoint is needed.

This research highlights a new perspective of reuse and describes improved retrieval techniques that decrease the associated investment costs, including those associated with traceability in the process, and reduces the chaos of ad-hoc reuse.

This new perspective, called Universal Knowledge Reuse (UKR), is fully integrated into the software development process. This methodology includes a set of tasks that tackle the problems of indexing, retrieval, and traceability along with the challenges of systematic and ad-hoc reuse. UKR also deals with the fact that reuse in the Information Age must be independent of the kind of information to reuse, the context where it must be reused, or even the user that demands it.

From our perspective, software reuse evolves towards knowledge reuse which necessitates a transition to Universal Knowledge Reuse.

An incremental, four-step experiment is included in order to validate the feasibility of this research and its application.
The first step estimates the costs of developing an indexer for one kind of information. The second step estimates the cost of changing an indexer in the case of an alteration to the corresponding information metamodel. The third step evaluates the relating indexers and the rates of retrieval for only one kind of information. The fourth step evaluates the capabilities of the UKR central environment in terms of cost and retrieval for a diverse set of information.

The structure of the document will be as follows:

- Section 1 contains a brief introduction to the content of the paper, including a summary of the research and its purpose.
- Section 2 presents the state of the art of the software reuse practice.
- Section 3 provides information to establish a common understanding of the research.
- Section 4 describes why it is important to move to knowledge reuse.
- Section 5 provides related research in the field of knowledge reuse or methodologies applied in this area.
- Section 6 presents the Universal Knowledge Reuse methodology.
- Section 7 describes the four-step experiment for assessing the hypothesis of this research.
- Section 8 reports the results found in each step of the experiment.
- Finally, the conclusions obtained after the study of the situation and the references cited are provided.

## 2 Software Reuse Practice

The driving principle of software reuse is to improve software production by reusing previously created assets (Frakes et al. 1998). The literature offers more than thirty different definitions for reuse, many of them collected in Llorens et al. (Llorens et al. 2006). Most definitions are operative guides for software re-users to develop new products by *finding* and *using* existing works/products developed for previous systems.

Due to historical reasons, most *systematic*, or formal, industrial reuse was initially based on domain engineering techniques (Karlsson 1995; Frakes et al. 2005; Neighbors 1994). Domain engineering based reuse models commonality and variability in order to define reusable assets. Engineers build reusable artifacts in advance by specifying their *commonality* in a domain and by defining adaptation procedures to allow those assets to deal with *variability*. The results of this process are pre-modeled assets: product lines, generators, frameworks, and so on. Once these assets have been implemented for reuse, they will be accessible to anyone who wants to use them. Karlsson has suggested a representation for the standard systematic reuse process (Karlsson 1995), combining software with reuse and software for reuse, as shown in figure 1.

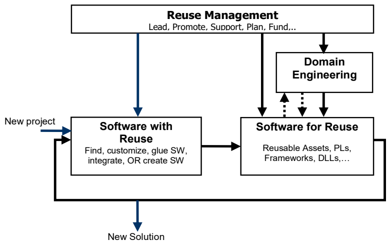


**Fig. 1:** Systematic Reuse Process representation, adapted from Karlsson (Karlsson 1995)

However, the major weakness of this approach was the substantial investment needed (Llorens et al. 2006). Reported low or negative return on investment (ROI) ratios became one of the main problems (Morisio 2002). As a result, software reuse did not achieve a stable position in the industry and started to decline.

In addition to systematic software reuse, individual practitioners have always reused code snippets, DLLs, components, and other simple artifacts. Public resources[1] provide reliable code snippets and components, usually for free (Hummel et al. 2008). This type of reuse is called *ad-hoc reuse*. There are many disadvantages of this non-systematic approach: (i) The practice is not integrated into the software development process due to its lack of formal foundations. (ii) It supports only code snippets, executables, and DLLs (not more abstract assets such as requirements and test cases). (iii) Retrieval capabilities (usually based only on text keywords queries) are relatively low. (iv) There is rarely any process for traceability as a reuse tool. For these reasons, ad-hoc reuse has very low rates of success.

New approaches try to solve the disadvantages of both types of reuse (systematic and ad-hoc) by focusing on the original goal of reuse: improving productivity. This can be achieved by removing requirements to perform domain engineering, as well as improving indexing, searching, and retrieval techniques of any type of asset on demand. These approaches would reduce the need for previously modeled assets such that ROI would not be a concern anymore and would allow for the transformation of the old ad-hoc practices into a new kind of systematic reuse. Software reuse involves not only code but also any type of information (requirements, risks, tests, models, manuals, etc).

---

[1] Some examples: http://souceforge.net, http://www.codeproject.com, http://www.codeplex.com/, http://www.google.com/codesearch, http://www.planet-source-code.com/, http://www.tucows.com/

Well-managed information is knowledge. This is the challenge that knowledge reuse faces today.

## 3 What is Knowledge?

Russel Ackoff (Ackoff et al. 1998), a systems theorist and professor of organizational change, developed the DIKW hierarchy with the levels Data, Information, Knowledge and Wisdom (originally called DIKUW because it included also Understanding). In this hierarchy still widely used by researchers, information is defined in terms of data; knowledge in terms of information; and wisdom in terms of knowledge. In particular, knowledge is defined as "application of data and information; answers 'how' questions."

Bellinger, Castro and Mills (Bellinger et al. 2004) further elaborated Ackoff's definition as follows: "knowledge is the appropriate collection of information, such that its [sic] intent is to be useful. Knowledge is a deterministic process. When someone "memorizes" information (as less-aspiring test-bound students often do), then they have amassed knowledge. This knowledge has useful meaning to them, but it does not provide for, in and of itself, an integration such as would infer further knowledge". This integration "requires a true cognitive and analytical ability that is only encompassed in the next level... understanding."

On the BITrum website, Pérez-Montoro summarizes diverse theories regarding knowledge (Dretske 1981; Floridi 2005; Semantic Conception of Information - Stanford; Gettier 1983; Hofkirchner 1999; Perez-Montoro 2007; Perez-Montoro 2004; Perez-Montoro 2001; Perez-Montoro et al. 2002; Sturgeon et al. 1998). One of the proposals is Floridi's semantic model (Floridi 2005), according to which "knowledge is constituted in terms of justifiable semantic information" and "information is the result of a data modeling process."

## 4 Why Knowledge Reuse?

We currently live in an information society where knowledge plays an unprecedented role in our personal and professional lives. Organizations have become conscious that their corporate information assets set them apart from their competitors. Thus, there is an increased interest in maximizing the value of an organization's knowledge; knowledge has become one of the most valuable assets of the modern organization (Davenport et al. 1998).

Knowledge achieved in one area may be applied in a different one. In order to utilize knowledge, one must find it and leverage it in the context of the new problem. Knowledge reuse is important for industry because it increases productivity by capitalizing on previous experiences and avoiding duplicated solutions. These benefits are, in short, the same well-known benefits (Karlsson 1995; Llorens et al. 2006; Poulin et al. 1993) of software reuse applied to knowledge in general.

It is difficult to think of an activity where reuse cannot be fruitfully adopted. A significant problem that companies have to manage is the variety of available knowledge. They are faced with the challenge of storing any kind of knowledge within a common repository, link the related knowledge elements, and finally offer reuse methods. Software production companies need to offer reuse of any kind of knowledge, in any context, at any time and to any user within the given access policies. Therefore, reuse must be considered as a service inherent to the knowledge management process. This concept of “reuse as a service” must be included in the definition of the reuse process.

Summing up, reuse is potentially beneficial in all domains and must be available as a service. Societies try to reuse energy, resources, artifacts, etc. In all aspects of life, individuals attempt to apply the knowledge gained from solving an older problem in the solving of a newer problem. Several applications of this idea for a single set of knowledge are described by authors Llorens et al. 2017; Fraga et al. 2015, 2016 and 2017; Chalé-Góngora 2017; Gallego at al. 2015; and Exman et al. 2015.

According to the Merriam Webster dictionary, the term ‘universal’ means characterizing or affecting everything, present everywhere, or adapted to meet varied requirements (Merriam Webster Dictionary, 2017). Therefore, reuse must be considered a universal concept, one that would benefit from a universal approach.

## 5 Related Work

The general knowledge reuse research domain was introduced by Markus M. Lynne. In his study *Toward A Theory of Knowledge Reuse: Types of Knowledge Reuse Situations and Factors in Reuse Success* (2001), Lynne presents the importance of knowledge within the industry and focuses on knowledge reuse studies according to the re-user and the purpose of reuse. It is important to find a knowledge reuse strategy at low cost as opposed to systematic reuse with negative or low return of investment. Mäki (2008) also showed that diverse kinds of information are related and should be linked in order to achieve better reuse results in terms of retrieval.

Related work regarding the reduction of cost in the reuse process and providing a methodology for reusing information was not found. Related research mainly focuses on knowledge reuse with or without the employment of ontologies, and is explained as follows:

### Exploring and Exploiting Knowledge: A Research on Knowledge Processes in Knowledge-Intensive Organizations

This PhD thesis developed by Eerikki Mäki (2008) at the Helsinki University of Technology investigates how knowledge is utilized, and is the most important source of competitive advantage for a growing number of companies and organizations. Knowledge workers predominantly work from, with, and for knowledge. While operations such as storing and transferring knowledge do not add value to knowledge as such, they are important for making knowledge available to members of an organization who need it. Results show that knowledge work is complex, and several challenges can be encountered when working with information and knowledge. Processes that connect dispersed knowledge and those that make it available to the members of an organization are highly interlinked.

The organizations studied operated with many types of knowledge each requiring a different set of management strategies. All of the organizations studied recognized the importance of encoding for making knowledge collectively available. As a result, the studied companies tried to increase the amount and quality of codified knowledge to improve the availability and reuse of information. Congruently, the studied organizations aimed at more routinized and formalized processes for managing knowledge. Even though a technology-based approach for managing encoded knowledge makes more sense than a human interaction-based approach, the studied companies had several problems in managing encoded information and knowledge. This study contributes to the understanding of knowledge work in general and helps explain how organizations can manage and reuse a diverse set of knowledge more effectively.

**Knowledge-Reuse for Innovation – The missing focus in Knowledge Management: Results of a case analysis at the Jet Propulsion Laboratory**

This research conducted in the Jet Propulsion Laboratory at the California Institute of Technology (Majchrzak et al., 2001) makes a distinction between the reuse of knowledge for routine tasks (e.g., use of templates, boilerplates, and existing solutions) and reuse that stimulates knowledge synthesis and innovation (e.g., searching a database to find new ideas to combine with existing knowledge). In this study, researchers derived a model that identifies eight factors likely to encourage knowledge reuse. From these eight factors, the researchers synthesized four generalizable factor categories in a variance model. In addition, the research yielded a process model that helps to explain how the eight factors influence knowledge reuse and how the reuse process unfolds in an innovation context. The study addresses knowledge reuse and the diverse set of interrelated factors but does not account for knowledge of any kind nor improve the process in terms of cost.

**Effects of Knowledge Reuse on the Spacecraft Development Process**

This master thesis developed by Esther Dutton (2008) introduces the problem of reusing knowledge for improving the spacecraft development process. The reusable assets must be created previously, but it considers any knowledge, but not any kind of it. After running simulations with a varied amount of reusable knowledge using different spacecraft of different complexities, the results showed that reusing knowledge decreases the spacecraft development time in complex projects, less complex project did not appear to benefit significantly from knowledge reuse. At the end the experiments showed that knowledge management is a worthwhile investment for the Jet propulsion Laboratory. The research focus on knowledge reuse in a specific context (spacecraft complex projects) but it lacks of facing the problem of cost or even the management of diverse kind of knowledge.

**Using Ontologies for Software Development Knowledge Reuse**

This research (Antunes et al., 2007) addresses software developers' need for better management techniques to handle increasing amounts of information and knowledge. This study focuses on the knowledge generated during the software development process. This knowledge can be a valuable asset for a software company, but it must be stored and managed for proper reuse. Ontologies are powerful mechanisms that can be used to model and represent knowledge, store it in a management system, and classify it according to the knowledge domain that the system supports. This work describes the Semantic Reuse System, which takes advantage of ontologies using the representation languages of the Semantic Web for software development knowledge reuse. It also describes how this knowledge is stored and the reasoning mechanisms that support reuse. Though the research focuses on knowledge reuse in the software development process context, it fails to address the problem of cost or the management of diverse kinds of information.

**KRAFT: Knowledge Fusion from Distributed Databases and Knowledge Bases**

This research (KRAFT, 2000) addresses the fact vital knowledge exists on the internet, but there is not a clear method for finding, adapting, and reusing it. Of course, the problem of finding information is not new, but automating the adaptation and reuse of knowledge for design applications presents a unique and difficult challenge. The KRAFT project is an exciting multi-site project which aims to tackle this problem by building a new kind of distributed information base. The information will be structured so that it can be adapted and transformed by intelligent processes running on various computers to make it more widely accessible. The project will also use multidatabase and object technology to seek out and select relevant knowledge. This contrasts with current internet usage that typically relies on tedious human intervention to click on icons and search by eye. This research is an interesting work offering the possibility of sharing structured information from diverse sources within a multidatabase, but it fails to address the problem of cost or even the management of diverse kinds of information. Though cost issues are not discussed, this is the most similar research to our own study.

**An Engineering Design Knowledge Reuse Methodology Using Process Modeling**

This research started by Baxter et al. (2007 and 2008) is an approach for reusing engineering design knowledge. Many design knowledge reuse systems focus exclusively on geometric data, which is often not applicable in early design stages. The proposed methodology provides an integrated framework bringing together elements of best practice reuse, design rationale capture and knowledge-based support in a single coherent framework. Rationale is supported by product information, which is retrieved through links to design process tasks. Knowledge-based methods are supported by a common data model, which serves as a single source of data to support the design process. Using the design process as the basis for knowledge structuring and retrieval serves the dual purpose of capturing and formalizing the rationale that underpins the design process and providing a framework through which design knowledge can be stored, retrieved and applied. The research proposes a methodology for reusing knowledge in early stages of process modeling, but again it does not address the problem of cost or the management of diverse kinds of information.

### CoMem: Knowledge Reuse From a Corporate Memory

This study (Demian et al., 2001) focuses on knowledge reuse from the personnel perspective, mainly in large companies. This research springs from the notion that "all design is redesign," meaning everything designed in the present is influenced by past designs. This idea can be extended to all phases of the life cycle of constructed facilities. Reusing knowledge from past experiences helps us design, construct, operate and maintain buildings more effectively.

This research also looks at how the process of knowledge reuse can be supported by a computer system. Knowledge is created when teams collaborate on projects. In this scenario, some of the knowledge generated ends up in the final product, but most of it remains in the memories of the team members. On the other hand, knowledge reuse can take the form of reflecting on personal past experiences or trying to decipher and understand knowledge captured by someone else. The aim is to capture all of that knowledge in a repository called the corporate memory so it can be reused.

Previous research projects have given a methodology for capturing knowledge in AEC projects. Knowledge management literature also yields the concept that ideas appreciate with usage, so a knowledge management system can act as a knowledge refinery. This final step of reuse is the focus of the research. The research proposes a step forward in knowledge reuse applied in the design field, but it does not address the problem of cost or the management of diverse kinds of information.

In summary, for all relevant studies shown, the main deficiencies are the study of cost and the representation of diverse kind of knowledge. Though contending with diverse kinds of information in knowledge reuse is not addressed in the research, it is a subject of interest because any organization must deal with diverse kinds of information at the end. The study of cost is also relevant; applying a proper methodology would be of value for any organization if it reduces cost.
(Lynne, 2001).

## 6 Towards a methodology for Universal Knowledge Reuse

Although the term "methodology" is used frequently in research, its meaning is not always clear. People usually do not distinguish between methodology, process, method and technique. For example, "methodology" is often erroneously considered a synonym of "process." In our view, a methodology is essentially a system of guidelines and can be thought of as the application of related processes, methods, and tools to a class of problems that all have something in common. In this paper we will use the following definitions, taken from Martin (1996) and Esteban (2008):

- A *process* is a logical sequence of tasks performed to achieve a particular objective.
- A *method* consists of techniques for performing a task; in other words, it defines the "how" of each task.
- A *tool* is an instrument that, when applied to a particular method, can enhance the efficiency of the task.
- An *environment* consists of the surroundings, i.e., the external objects, conditions, or factors that influence the actions of an object, individual, or group. These conditions can be social, cultural, personal, physical, organizational, or functional.

Figure 2 illustrates the relationships between the so-called PMTE elements (Process, Methods, Tools, and Environment) along with the effects of technology and people on them.

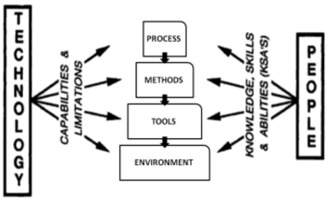


**Fig. 2:** PMTE elements and the effects of technology and people (Martin 1996).

As Martin explains (1996), "the capabilities and limitations of technology must be considered when developing the software engineering development environment. Technology should not be used just for the sake of technology. When choosing the right mix of PMTE elements, one must consider the knowledge, skills and abilities (KSA) of the people involved. When new PMTE elements are used, often the KSAs of the people must be enhanced through special training or special assignments… Buying expensive tools without providing training to use them most often does not improve the development process, since the tools may be misused, or not used at all." Referring to the use of technology in methodologies, Martin states that buying tools just to comply to a checklist or without training makes no sense. Doing so will incur extras costs in the process due to lack of expertise, involuntary errors, or even underuse of technology.

According to the definition made by Martin (1996), the Universal Knowledge Reuse Methodology is a collection of processes, methods, and tools associated with an environment of technology and people, with the aim of reusing any kind of knowledge from its content and syntactical structure.

The Universal Knowledge Reuse Process (Figure 3) follows the incremental approach introduced by Llorens et al., (2006) where each task must be incrementally reviewed and maintained. Knowledge is not static; rather, it evolves with the domain.

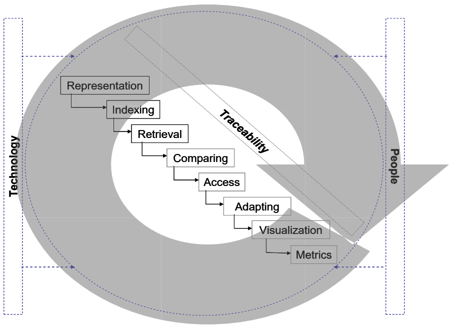


**Fig. 3:** Universal Knowledge Reuse Process (Fraga 2010).

The idea behind Universal Knowledge Reuse is that knowledge is one of the main assets humans possess. Knowledge achieved in one area may be applied in a different area if it is memorized and taken advantage of in a new area or problem. The application of computer tools and techniques simulating human memorization and application of knowledge could greatly enhance the power of knowledge reuse.

For these reasons, the knowledge reuse process needs to accommodate knowledge of any type, used by anyone and in any situation.

UKR complies with this universality, but its application in a technological environment may be different (e.g., dependent upon the type of operating system in use). Some of the challenges that must be faced are: knowledge representation; access, retrieval and adaptation for reuse; information extraction; implementation of reuse techniques in diverse environments; and reuse metrics.

The Universal Knowledge Reuse Process involves different tasks than systematic reuse. The UKRM tasks are Representation, Indexing, Retrieval, Comparing, Access, Adapting, Visualization, Metrics and Traceability. Traceability takes precedence over all the other tasks, as tracing how elements are changed and how they relate to other elements is key for reuse. These tasks are more or less present in systematic reuse, but need to be analyzed and generalized in order to manage any kind of information, by any user in any context (thus the motto “anything, anyone, anywhere”).

## Universal Representation of Knowledge

In order to achieve the reuse of any kind of knowledge, it is necessary to find an organizational repository that can store various types of knowledge assets, regardless of their syntactical structure. The Knowledge Organization Systems (KOS) is a formal and well-studied domain for creating accurately structured knowledge, where ontologies have a fundamental role (KOS; Bechofer et al. 2001; Hill et al. 2002; Janée et al. 2002; Frakes et al. 2005). Ontologies play a major role in UKRM, mainly as a guide for storing semantic data within repositories such that they can become reusable assets.

Many types of schemas for representing information are available (Davis et al. 1993; Brachman 1983; Sowa. 2000; Davenport et al. 1998; Abecker et al. 1998; Kuhn et al. 1997; Buckingham 1998; Sanchez-Cuadrado et al. 2007). UKRM has selected RSHP due to its capability to deal with any kind of syntax and content of artifacts (Llorens et al. 2004; Llorens et al. 2006). RSHP, short for *RelationSHiP*, is based on the idea that knowledge is, in essence, a set of related concepts. Therefore, it should be possible to represent any kind of knowledge as relationships between ontological concepts.

## Universal Indexing

The universal indexing process in UKRM is a process of transforming various kinds of knowledge into a format compatible with a universal representation model. Doing this without losing information is difficult both in current systems and in the UKRM process.

UKRM indexing process deals with this problem as follows:

- The transformation rules use the metamodel (i.e., the syntactical structure) of the knowledge to be represented; if this metamodel is unknown, then it must be inferred from the content. The metamodel provides the structure of the representation language and makes it possible to deal with the knowledge in a higher level of abstraction. A suitable language for representing the metamodel is the Unified Modeling Language (UML) (Fowler 2003) (UML Standard specification - OMG).
- The creation of transformation rules that make the process automatic, using the information and its metamodel as input.

All UKRM activities (Figure 3) require dealing with all kinds of knowledge equally; therefore they must work at the metamodel level.

Currently each kind of information needs a specific indexer for extracting and accumulating the information. Universal indexing provides a unique configurable indexer (Figure 4).

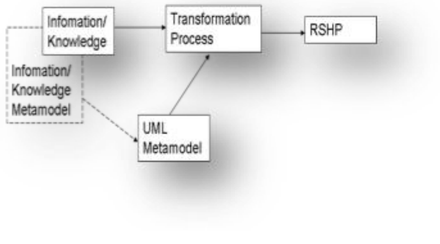


**Fig. 4:** The indexing process for many types of information using a universal indexer supported by the information's metamodel.

**Universal Retrieval**

Universal retrieval must be independent of the type of knowledge. RSHP includes a universal retrieval model based on artifact content instead of keywords. This aids in the search of artifacts by content likeness and not only by description. RSHP provides powerful retrieval capabilities based on graph representation for relationships and concepts.

**Universal Comparing**

As mentioned before, transferring knowledge is an activity intrinsic to the reuse process, because the goal of reuse is to take advantage of retrieved knowledge in the context of new areas or problems. In the UKRM, this activity can be accomplished through the application of the universal comparison of knowledge. UKRM offers reusers a set of automatic procedures that compare the artifacts' contents to aid in the process of selecting reusable contents. The RSHP universal schema offers the ability to compare concepts and relationships within the representation model, so this activity is solved at the RSHP side.

**Visualizing Knowledge**

Since knowledge can be of any kind, reusers must be provided with a universal visualization tool. Because indexers must represent all content for all artifacts within the repository, a general visualization process at the representation metamodel level provides a universal representation. RSHP has a visual mode based on graphs that aids in this activity.

**Universal Adapting**

After knowledge is retrieved and visualized, it has to be transferred for use in a new context. Universal copy/paste deals with the copying and pasting of any kind of knowledge, in any context, within the appropriate rights of the user. Universal traceability (i.e., the availability of a trace between any kinds of assets at the moment of copying/pasting) is part of the universal copy/paste process.

**Universal Metrics**

Classical empirical studies in the software reuse domain are based on measuring software reuse, and code reuse in particular (Poulin et al., 1993). Therefore, the whole process must be measured or assessed after reuse. In UKRM, measurement is not only performed for code reuse, but for the entire reuse process. Universal metrics deal with measuring any kind of reused knowledge. For this purpose, the users must create new metrics.

**Universal Access**

In order to offer any kind of knowledge to any re-user, UKRM provides a universal access procedure. Knowledge can be extracted from any software asset and reused for any other software asset. Reusers only have to select or define the meta-model that structures the knowledge to be reused. In order to allow this functionality, UKRM proposes the reuse contract, a basic unit that links a piece of knowledge with its metamodel; the provider (e.g., a window); the user; the rights; and any additional information. The user is the agent in charge of describing this contract depending on his or her needs.

**Universal Traceability**

The trace must be present in all stages of the process. Any artifact can be traced against another. This is handled in the RSHP universal schema.

## 7 Experiment Step by Step

In order to validate the research, a four-step experiment was designed and performed. The experiment focused on the main hypothesis: *is it possible to reuse any kind of information at low cost?* The experiment was intended to prove the feasibility of the hypothesis, but also to improve the solution developed for the experiment. The software for the test was developed in C#.NET. A description of each step of the experiment is shown below, with special detail provided for the third and fourth steps in particular, as these steps include retrieval and cost estimation.

In the case of the experiment, the corpus built for one kind of information and for diverse kinds of information is available in the research group homepage (http://www.kr.inf.uc3m.es/es/investiga-con-nosotros/miembros-del-groupo?view=thesis&task=show&id=2)

### Step 1

The first step *estimated the costs of developing an indexer for one kind of information*, and for the universal indexer in particular. In this step, development costs of the universal indexer, a generic indexer, a commercial indexer, and an ad-hoc indexer were compared. The costs were estimated using an earlier stage estimation method based on use cases. The first step focused on testing the cost of developing different indexers for one kind of information.

Several methods are available for estimating the cost of development. We selected an estimation model based on Use Case Points as formulated in [8], and Enterprise Architect 7.5 as the modeling tool to create the use cases.

### Step 2

The second step *estimates the cost of change for an indexer in the case of an alteration to the corresponding information metamodel*. For this step, a survey was created to acquire real information from companies regarding the development of new releases due to changes in the requirements and structure of the information. This step focused on evaluating the evolution of indexers when a change on the metamodel occurred. In that case, an incremental cost of developing must be added.

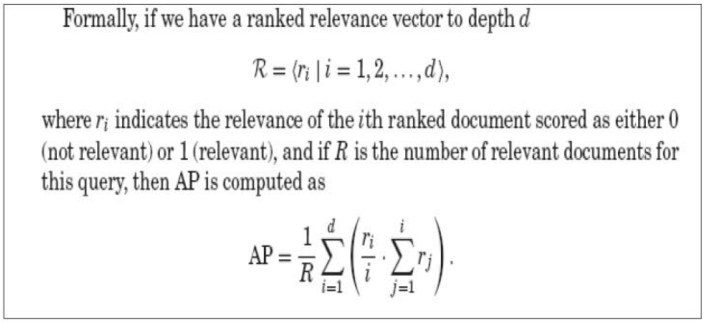
### Step 3

The third step *estimates the whole environment and process, including the indexer, retrieval, and repository in terms of retrieval parameters*. The universal indexer performance is compared to the rest of the indexers studied. Average Precision (AP) was chosen as a good and simple option to measure retrieval in this experiment (Turpin et al. 2006).

This step focused on *estimating the retrieval for one kind of information, relating indexers, and the rates of retrieval achieved*.

### Design and accomplishment of the third step of the experiment in detail

1. Selection of a metric for retrieval analysis:
In the information retrieval systems for each document in a collection, a score that estimates the similarity between that document and a query is computed (Moffat, 2008). In typical systems, each score represents an estimated probability that the document is relevant to the information need expressed by the query. Two elementary measures are recall and precision (Baeza-Yates et al., 1999) (Manning et al., 2008) (Salton et al., 1983). These can be united to give a single value via mechanisms such as 3-point or 11-point recall-precision averages (Buckley and Voorhees 2005). One commonly used measure in recent IR research is mean average precision (AP), which does not directly use recall, but does require knowledge of R (i.e., the total number of relevant documents for the query in question). In this experiment, AP will not fail because of inflated effectiveness estimates due to incomplete relevance judgments; the relevance judgments are complete because the Corpus indexed is well known. For that reason, AP was a good and simple option for measuring retrieval in the experiment. The formal definition is shown in the following figure (Turpin et al., 2006).



Formally, if we have a ranked relevance vector to depth $d$

$$\mathcal{R} = \langle r_i \mid i = 1, 2, \ldots, d \rangle,$$

where $r_i$ indicates the relevance of the $i$th ranked document scored as either 0 (not relevant) or 1 (relevant), and if $R$ is the number of relevant documents for this query, then AP is computed as

$$\mathrm{AP} = \frac{1}{R} \sum_{i=1}^{d} \left( \frac{r_i}{i} \cdot \sum_{j=1}^{i} r_j \right).$$

**Fig. 5.** AP formal representation.

While using the AP method for evaluating retrieval, a matrix of relevance for all documents in the corpus was created and made available in the repository of the experiment.

2. Corpus construction:
In this step of the experiment, a corpus was built for testing retrieval capabilities. For that purpose, the kind of information must first be established. Secondly, the set of documents must be selected and organized. The information used in this step was the structured information derived from class diagrams articulated in XML Metadata Interchange (XMI), a definition made by OMG standards.

The Corpus consists of more than one hundred diagrams downloaded from the internet from the first 150 pages of a Google image search using the keywords “class diagram.” The images were selected according to the following criteria:

- Legibility
- Images actually corresponding to class diagrams
- Semantic information available

Using the functionality of Poseidon software that allows diagrams to be exported in XMI format, an XMI version of each of diagram in the corpus was generated, creating the XMI Corpus. The corpus for indexing handles information in the following four domains: education, code classes, animals, and game environments. It is interesting to note these are the main domains found when searching on Google for creating the corpus. Furthermore, an ontology for the four domains is included with relevant information of synonyms and special associations useful for retrieval purposes.

3. Queries definition:
The queries for searching in the created corpus dealt with the domains found, along with queries for the following categories:

- Keyword: a query based only on simple keywords. The documents returned must contain the keywords entered.
- Semantic: a query based on a relationship between two keywords. The results returned must be documents containing a relationship between the selected terms.

- Strong Semantic: a query based on a relationship between more than two keywords. The results returned must be documents containing a relationship between the selected terms.
- Calculated: a query based on calculated values. It must return a number calculated or a single term resulting from a query based on relationships and keywords.

4. Retrieval Analysis for an ad-hoc indexer for one kind of information:
In the case of an ad-hoc indexer, results of any category of query will be one hundred percent retrievable because the software must commit all user requirements by all means, including functional and non-functional requirements. An ad-hoc software was developed by and for the user in order to accomplish all needs.

5. Retrieval Analysis for a commercial indexer for one kind of information:
After evaluating the most popular commercial software available for managing the class diagrams as information and its metamodel (Chitnis, 2003) (Poseidon for UML, Altova Modeler for UML, Enterprise Architect 7.5, Rational Rose), the retrieval capabilities of the commercial indexer were determined to be based in keyword queries, not by relationships at all.

6. Retrieval Analysis for a universal indexer for one kind of information:
The corpus was indexed using the universal indexer developed for the core universal knowledge reuser; a repository full of indexed information was the result. The repository was then used by the retrieval software to apply the queries previously formed.

7. Retrieval Analysis for a generic indexer for one kind of information:
In the case of generic indexers like Google Desktop or Windows Search Engine, queries are based on keywords only. Because the corpus and organizational information was located in directories, the generic indexer selected was the Windows Search Engine for its ability to set retrieval by directory.

### Step 4

Finally, the fourth step *evaluated the capabilities of the Universal Knowledge Reuse central environment in terms of cost and retrieval for diverse kind of information, not only one kind of information as evaluated in the previous steps*. This step proved the advantage of retrieving content-based structures from diverse kinds of information. Lastly, an additional study was conducted to validate the use case estimation method.

### Design and accomplishment of the fourth step in detail

1. Selection of a metric for retrieval analysis:
For consistency and comparison, the same retrieval metric used in the previous step was used in this final step.

2. Corpus construction:
This step necessitated choosing diverse kinds of structured information metamodels, as the following diverse domains were selected: Medicine, E-commerce, Mathematics, Images and Knowledge Organization Systems. For these domains, some of the available metamodels were MEDLINE for Medicine; EDI for E-commerce; ECML for E-commerce; MATHML for Mathematics; MIX for Images; THESAURI for KOS (Thesauri); OWL for KOS (Ontologies); and XTM for KOS (Topic Maps).

The metamodels used in this research for each kind of information were formalized in Document Type Definition (DTD) or XML Schema. Sets of documents complying with every single metamodel were selected from the internet. Available university repositories were also used to acquire more documents.

After that, the metamodels were treated for semantically related information so the richness of the retrieval could be tested when linking diverse documents and metamodels. For instance, if Annabel created an image of Stockholm and also wrote an article in a medical journal in Stockholm, the information must be reflected in two documents: one related to pictures (Mix) and another related to medical journals (Medline).

When searching for “Annabel related to Stockholm” or “Anabel Madrid” both documents must appear as relevant. An important part of the richness of using the universal indexer is that the documents must be located if the information is available, regardless of context.

A user could reuse the resulting information in any of the remaining activities proposed in the Universal Knowledge Reuse Process; for instance, the universal copy/paste can be used for adapting the information in any context.

Furthermore, an ontology for the domains represented is included with relevant information of synonyms and special associations among terms to aid in retrieval.

3. Cost Analysis of change for each kind of information indexer:
Here, it is important to note the quantity of change needed when new requirements related to the metamodel structure demanded change. The *easy, medium* and *high* levels of cost change due to customizing the universal indexer are shown as follows:

**UI Change cost (Euro)**

| | |
|---|---:|
| ***Easy*** | 4800 |
| ***Med*** | 6400 |
| ***High*** | 8000 |

An *easy change* refers to a change in new namespaces; a *medium change* signifies a change in the attributes or relationships; and a *difficult change* denotes a change in code because of customization of element recognition. The costs were a result of the analysis of the Use Case Points method and assumed the complexity for the Use Case change was simple, medium or high. For the selected metamodels, the change needed was not difficult, so it was assessed as easy.

In this step, an additional study was conducted to compare the Use Case estimation method with a real development. For that purpose, a Final Career Project was created and a student was asked to develop a topic map management tool with capabilities of searching. The estimated cost for this kind of information and the real cost is compared in the result analysis. Even though the cost for developing each of the indexers was estimated, the data in the result section of this study is relevant for comparisons.

4. Queries definition:
The queries for searching within the corpus using diverse kinds of information metamodels had to contend with the domains and the content itself. Queries were simple, based on keywords and semantic relationships as follows:

- For Query 1, Annabel sent orders to Stockholm and made pictures of Stockholm.
- For Query 2, the search terms "abdomen" and "body parts" were related in two topic map files from different domains: one for Medicinal Plants and one for Health Care.
- For Query 3, AFV (synonymous for Annabel) wrote medical records in 1980.
- For Query 4, Annabel, produced the same file resulting from Query 3.
- For Query 5, simple search keywords were used to search amongst different kinds of information. For instance, the word "infinity" was used in both a metamodel file and a content file.

## 8 Global results

### Step 1 results

In the first step, as shown in Figure 6, the most expensive developments were the universal indexer and the ad-hoc indexer. The least expensive developments were the desktop search indexer and the commercial indexer.

When the metamodel of the information kind changes, the evolution of cost must be analyzed. According to our study, the more competitive indexers were the desktop search indexer and the commercial indexer. These results are premature, however, as the retrieval results must be also analyzed.

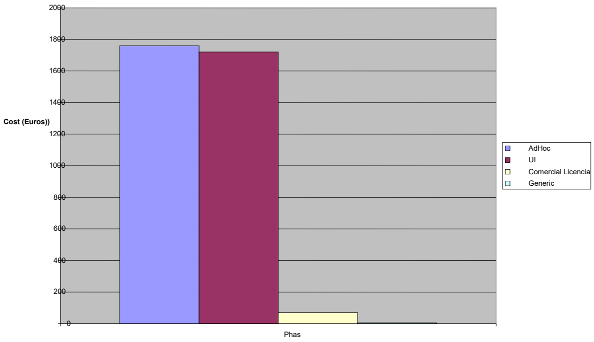


**Fig 6:** Cost of development for each indexer.

### Step 2 results

In order to understand and evaluate the real market cost proportions for new releases of prior developments, an anonymous survey for companies was created. The survey contained several questions about cost, percentages of cost, and both minor and major changes in software development. The survey was designed to capture the following information:

- Company size
- Countries where the business is deployed (national or international)
- Years in the market
- Number of employees
- Quantity of releases developed
- Compatibility between releases
- Investment in new versions
- Certifications for quality measures achieved
- Information systems integration cost
- Causes of change
- Implications of minor changes
- Implications of major changes

The companies contacted were widely distributed in size and countries of operation. Companies were sent custom e-mails in order to obtain better results in terms of survey response. The survey was available in Word format and PDF (Portable Document Format), by FTP (File Transfer Protocol), or by e-mail to satisfy the preference of the individuals contacted. Around two hundred people were contacted, and the names of the companies are not provided due to anonymous participation. The survey questions regarded the kind of change that took place (minor or major), and the percentage change a new requirement involved depending on its kind. The survey and its results will be published in an additional publication due to their implications for and relevance to the research and industrial worlds.

The cost of developing diverse indexers for one kind of information when a change in the metamodel is needed is shown in Figure 7.

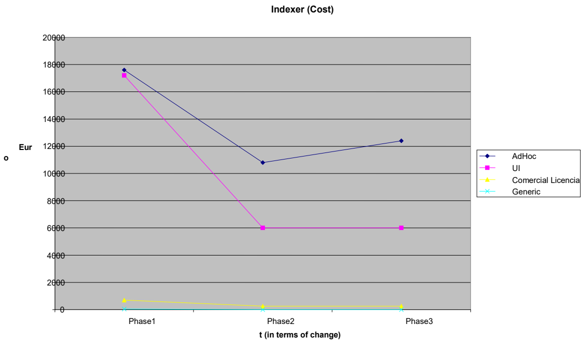


**Fig 7:** Behavioral tendency of costs.

As shown in Figure 7, the tendency of the cost of change for the ad-hoc indexer was to grow constantly. The tendency for UI cost was to remain more or less constant depending on the customization needed.

Adaptation to a given change in the metamodel was not suitable for the generic desktop indexer, meaning there was no cost associated with metamodel changes. However, the change was not reflected because the philosophy of these generic desktop indexers is keyword based.

By using Statgraphic Centurion XV statistical software for analysis of data (extending the available information), the sequence plots of the four cost variables for each indexer are shown in Figures 8-11.

A simple regression could be done between the commercial and Google desktop indexers, the commercial and universal indexers, and the Google desktop and universal indexers because the P-value is less than 0.05. In those cases, a linear model describes the relationship between the variables. A study was done to show the correlation between the commercial costs and the desktop costs. In this case, the dependent variable is commercial and the independent variable is desktop.

Using a linear model Y = a + b*X, the coefficients are:

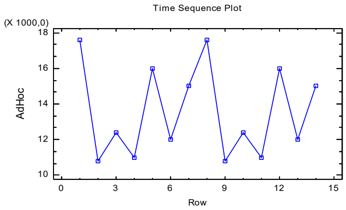


**Fig. 8:** Sequence Plot for Ad-Hoc Costs.

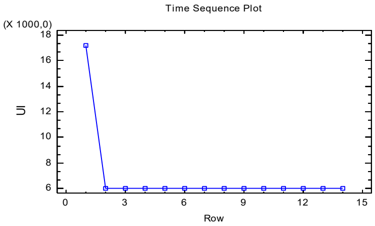


**Fig. 9:** Sequence Plot for UI Costs.

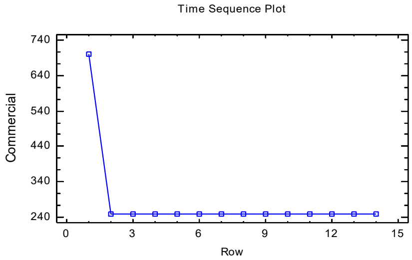


**Fig. 10:** Sequence Plot for Commercial Costs.

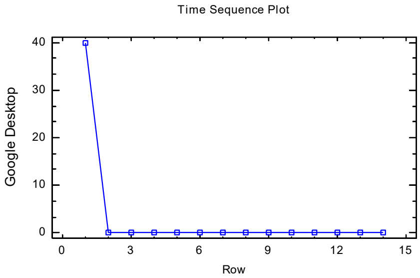


**Fig. 11:** Sequence Plot for Generic Desktop Costs.

| | Least Squares | Standard |
|---|---|---|
| Parameter | Estimate | Error |
| Intercept | 249.0 | 0.0 |
| Slope | 11.25 | 0.0 |

**Table 1:** Linear model coefficients.

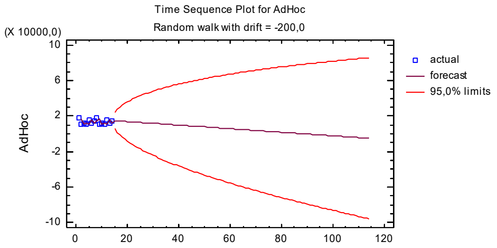


**Fig. 12:** Forecast for Ad-Hoc Costs.

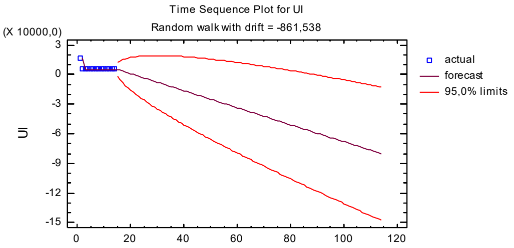


**Fig. 13:** Forecast for UI Costs.

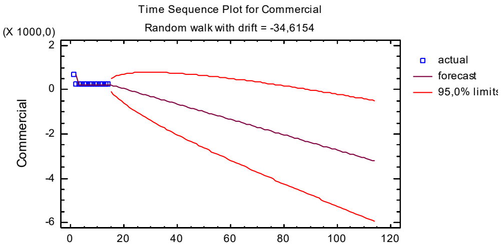


**Fig. 14:** Forecast Plot for Commercial Costs.

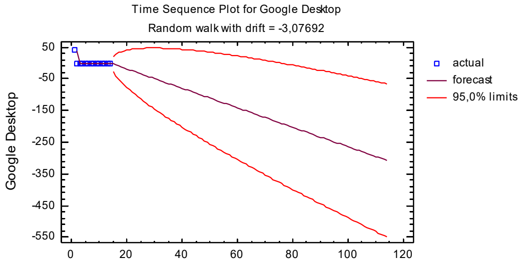


**Fig. 15:** Forecast Plot for Generic Desktop Costs.

And the analyses of variances are:

| Source | Sum of Squares | Df | Mean Square |
|---|---|---|---|
| Model | 188036.0 | 1 | 188036.0 |
| Residual | 0.0 | 12 | 0.0 |
| Total (Corr.) | 188036.0 | 13 | |

**Table 2:** Variances.

The Correlation Coefficient was 1,0; R-squared was 100,0 percent; R-squared (adjusted for d.f.) was 100,0 percent; Standard Error of Est. was 0,0; Mean absolute error was 0,0; and Durbin-Watson statistic was 1,0 (P=0,0141).

The output shows the results of using a linear model to describe the relationship between commercial and Google desktop costs. Both samples came from normal distributions. The equation of the fitted model is:
commercial = 249 + 11,25*desktop

The R-squared statistic indicates the model as fitted explains 100,0% of the commercial variability. The correlation coefficient equals 1, indicating a relatively strong relationship between the variables. Since the P-value is less than 0,05, there is an indication of possible serial correlation at the 95,0% confidence level, but correlation does not imply causation. Furthermore, the residuals could be tested using the Durbin-Watson (DW) statistic tests to determine if there is any significant correlation based on the order in which they occur in the data file.

At this time, if we analyze the survey results regarding metamodel changes, the response received from completed surveys was forty out of two-hundred (20%). The most typical negative responses were:

a) The respondent does not have the information
b) The respondent works in consultancy or technical positions and did not have an idea of the percentages of cost for new releases
c) The respondent has the information but did not measure it
d) The respondent cannot share this information

It is interesting to note most of the companies considered a change in the information metamodel to be a major change. The cost of minor changes ranged from 1% to 30% while the cost of major changes ranged from 40% to 100%. The size and nationality of the companies was not relevant when considering releases and new developments.

Given that, the conservative model followed here for the estimated costs of changes would be even more superior in the commercial world if we consider the change in the metamodel to be a major change.

Finally, it is interesting to note most of the companies make a change in software only if clients ask for it and pay for it; there are no altruistic companies so far. A complete study and analysis of the responses was a tangential study opened by this research. The result will be available for companies and researchers in a further publication.

In summary, the costs of the universal, desktop, and commercial indexers tended to drop while the cost of the ad-hoc indexer tended to remain constant over time.

The results of this step separate the UI cost and the ad-hoc cost that were quite level in the first step of the experiment. Comparatively, this means the UI has an advantage over the ad-hoc software because costs will be reduced over time.

**Step 3 results**

The behavior followed by each system in the retrieval task is shown as follows:

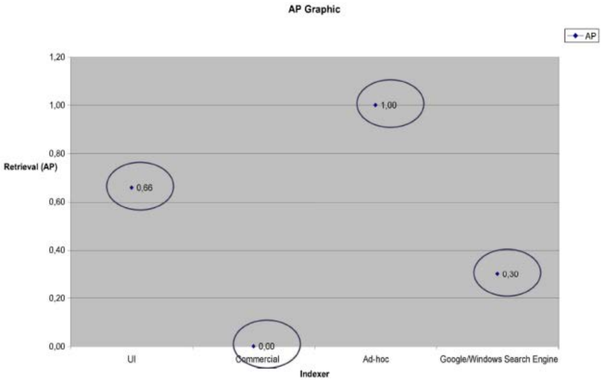


**Fig. 16:** AP Graphics behavior for each indexer.

The ad-hoc software exhibited the best result in terms of retrieval, followed by the UI and the generic software. Though the UI achieved a good AP factor, it was inferior to that of the ad-hoc indexer.

Next, productivity was used to measure the results achieved in the experiment compared to the resources used as input. The productivity needed to be calculated for both variables simultaneously, as cost and retrieval capabilities affect investments and results. This concept is used broadly in industrial engineering. The most general concept of productivity is (Jorgenson et al. 1967) (Sumanth 1979) (Brayton 1983):

**Productivity = Output = Results Achieved Using the Input Resources**

Or in this case:

**Productivity = Results/Inputs.**

The cumulative graphical behavior of the normalized data (in scale 1:100000) is shown as follows (proprietary = commercial):

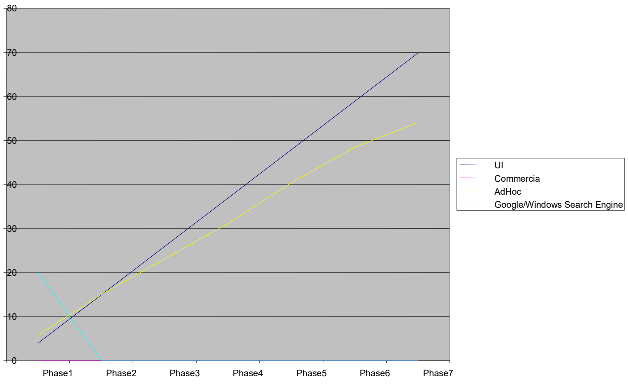


**Fig. 17:** Cumulative graphical behavior of productivity by indexer.

Retrieval for the generic indexer was only satisfactory in the case of keywords. For the proprietary indexer, productivity was null because it produced no results.

In the case of the ad-hoc indexer and the UI, the proprietary tendency was to decrease by the same amount that the change phases grew, and the UI productivity tendency grew in a linear manner.

The forecast showed an increase in productivity for the UI as compared to a decrease in productivity for the ad-hoc indexer; in other words, there was an inversely proportional relationship between the two variables.

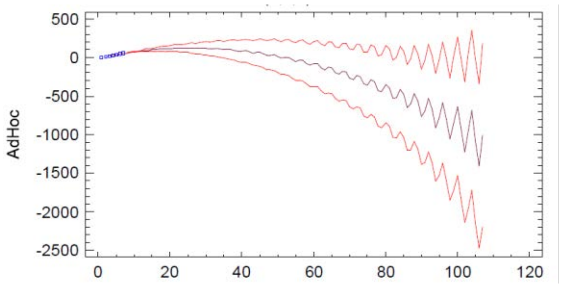


**Fig. 18:** Cumulative productivity forecast for ad-hoc variable: forecast in dark color and current in red with limits of 95%.

Summing up, the cumulative productivity forecast showed the UI represents a comparative advantage to the ad-hoc indexer even though their respective costs were similar in the results of the first experiment.

### Step 4 results

The fourth step focused on retrieval and cost for diverse kinds of information using various structured information metamodels. The domains selected in this step were Medicine, E-commerce, Mathematics, Images and Knowledge Organization Systems. Some of the available metamodels for these domains were MEDLINE for Medicine; EDI for E-commerce; ECML for E-commerce; MATHML for Mathematics; MIX for Images; THESAURI for KOS (Thesauri); OWL for KOS (Ontologies); and XTM for KOS (Topic Maps). These metamodels are available at the following link with the corpus and the software implemented:

(http://www.kr.inf.uc3m.es/es/investiga-con-nosotros/miembros-del-groupo?view=thesis&task=show&id=2)

Sets of documents complying with every metamodel were selected from the internet and additional documents were acquired from available university repositories. After that, they were treated for semantically-related information amongst diverse documents so the richness of the retrieval could be tested. For instance, if Annabel created an image of Stockholm and also wrote an article in a medical journal in Stockholm, the information must be reflected on two documents: one related to pictures (Mix) and another related to medical journals (Medline). When searching for "Annabel related to Stockholm" or "Annabel Stockholm," both documents must appear as relevant. This feature is key to the richness of universal indexer; the documents containing the relevant information must be retrieved regardless of context. Furthermore, an ontology for the domains represented was included with relevant information of synonyms and special associations among terms useful for retrieval purposes.

The cost resulting from the parallel development of the software is:

| | **Amount (Euros)** |
|---|---|
| Personnel | 9030,00 |
| Equipment | 128,62 |
| Licenses | 1356,00 |
| Travel | 488,00 |
| Other | 414,00 |
| **Sub-Total** | **11416,62** |
| Benefit (20%) | 2283,32 |
| Risk (15%) | 1712,49 |
| TAX (16%) | 1826,66 |
| **Total** | **17239,10** |

**Table 3:** Parallel development.

| | |
|---|---|
| Costs of parallel development | 17239,10 |
| Estimated cost using UCP | 16000,00 |
| **Difference (Euros)** | **1239,10** |
| **Difference (%)** | **4** |

**Table 4:** Summary of costs.

The resulting cost was likely to be similar to the cost estimated using UCP. In this case, the difference was only 4%, meaning the estimated cost was not far from reality and was in accordance with the estimation method error percentage.

The cumulative cost for diverse kinds of information was calculated as shown in Figure 19.

The UI cost for each kind of information grew less than the estimated cost for each indexer when cumulative. The cost for customizing the UI indexer for one kind of information was negligible compared with the cost of developing an indexer for each kind of information.

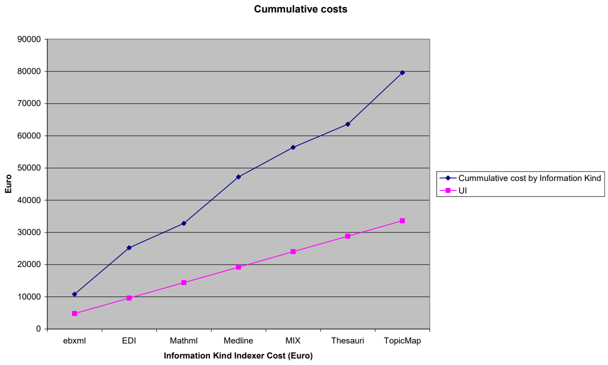


**Fig. 19:** Cumulative cost of information kind costs and UI costs.

The statistical behavior for the cumulative cost of information kinds and UI costs is shown as follows, as well as the forecast for each variable. The forecast showed an increase in productivity with a decrease in cost for the UI.

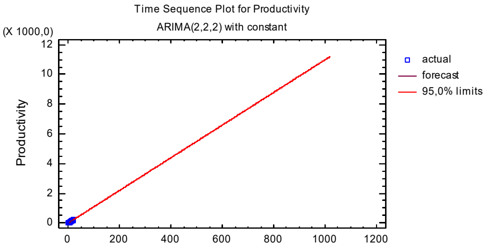


**Fig. 20:** Cumulative productivity forecast for diverse kinds of information.

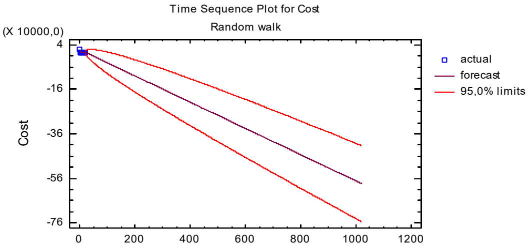


**Fig. 21:** UI Costs.

The cost follows a descending pattern in the forecast, as shown in Figure 21.

The Correlation Coefficient is -0,553492; R-squared is 30,6353 percent; Standard Error of Est. is 56,778. An approximate 95% of confidence interval for power: -0,226 to 1,037. The Power is 0,371 and the Shift is 0,0.

This procedure was designed to allow comparison of the effect of various power transformations of productivity on the linear regression between productivity and cost. Shown as a solid blue line in Figure 22, the equation of the fitted model was:

BoxCox(Productivity) = 280,479 - 0,014665*Cost, where:
BoxCox(Productivity) = 1 + (Productivity^0,371-1)/(0,371*79,04^-0,629)

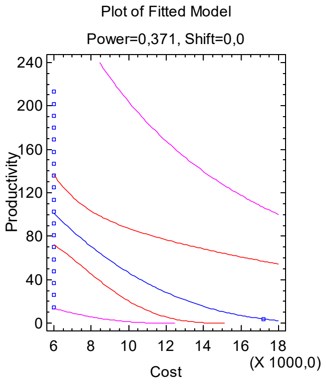


**Fig. 22:** Box-Cox plot.

This Box-Cox transformation had the power determined in order to minimize the mean squared error (MSE).

Since the P-value in the ANOVA table is less than 0,05, there is a statistically significant relationship between the transformed values of productivity and cost at the 95% confidence level. The R-squared statistic indicates the model as fitted explained 30,6353% of the variability in productivity. The correlation coefficient equals -0,553492, indicating a moderately strong relationship between the variables. The standard error of the estimate shows the standard deviation of the residuals to be 56,7786. This value can be used to construct prediction limits for new observations.

In summary, the productivity demonstrated by UI for diverse kinds of information was higher than the one shown in the previous step of the experiment. The greater the variability of the kinds of information, the more productive our process will be.

The feasibility of the UKRM process based on its core methods has been proved. Furthermore, the productivity it provides is superior to that of the other tested.

## Conclusions

Acceptance of the classical systematic reuse process has been hampered within the industrial environment because of the huge investment required. Low or negative ROI ratios became a main hindrance when organizations failed to reuse a huge number of components developed for reuse. On the other hand, ad-hoc reusers gained a certain level of success but with a low level of accomplishment; the practice of ad-hoc reuse was rather chaotic and only applied to code fragments, DLLs and full components.

Industry would benefit from the ability to reuse any kind of knowledge, in any context, and by any user. We have presented the concept of the Universal Knowledge Reuse methodology (UKRM) so that others can enjoy the benefits of theoretical reuse applied to any kind of knowledge within software domains and avoid the well-known drawbacks of systematic and ad-hoc reuse.

The UKR process is based on a universal representation model, a universal indexer, and universal activities such as retrieval, transferring, access, knowledge visualization and reuse metrics. All of these activities have been designed in UKRM so that they can be applied to any kind of knowledge, in any context, for any user. The methodology is universal in terms of context, but in terms of technology, universality depends on where the methodology will be applied. The UKR representation model and associated graphical representations have been improved from previous versions so that they can fully support the UKR process and methodology.

We have conducted an experiment to measure the effectiveness of UKR based on ROI, cost estimation techniques and retrieval results. The experiment revealed a positive effect for the application of UKR in the industrial world. The reduction of effort and investment associated with the reuse of previously developed documents, code and even software models proves UKRM is a powerful asset to be integrated into the software development process. The results of reduced costs with acceptable retrieval ratios show the UKRM core activities yield greater productivity compared to other indexing-retrieval systems. Comparing investments versus results achieved, we conclude that Universal Knowledge Reuse core tasks behave better than the rest. The results prove that reuse at low cost is possible and furthermore, its productivity is enhanced in the case of any kind of knowledge. In short, productive and low-cost Universal Knowledge Reuse, based on universal knowledge indexing and retrieval, is feasible.

## References

Abecker, A., Bernardi, A., Hinkelmann, K., Kuhn, O. and Sintek, M. (1998), Towards a Technology for Organisational Memories, IEEE Intelligent Systems, 13(3) pp. 30-34.

Ackoff, R. L., "From Data to Wisdom", Journal of Applied Systems Analysis, Volume 16, 1989 p 3-9.

Antunes, B. , Seco, N., Gomes, P. Using Ontologies for Software Development Knowledge Reuse. Lecture Notes in Computer Science. Volume 4874/2007. 2007.

Baeza-Yates, R., et al. Modern Information Retrieval. Addison Wesley Longman Limited. 1999.

Baxter, D. et al. An engineering design knowledge reuse methodology using process modelling. Research in Engineering Design 18 (1) pp. 37-48. (2007)

Baxter, D., Gao, J., Roy, R. Design Process Knowledge Reuse Challenges and Issues. Computer-Aided Design & Applications, 5(6), 942-952. 2008.

Bechofer S., Goble C. Thesaurus construction through knowledge representation. Data & Knowledge Engineering, 37, 25-45. 2001.

Bellinger, G., Castro, D., Mills, A. Data, Information, Knowledge, and Wisdom. http://www.systems-thinking.org/dikw/dikw.htm (2004)

Brayton, G.N. (February 1983). Simplified Method of Measuring Productivity Identifies Opportunities for Increasing It. Industrial Engineering.

Buckingham Shum, S, (1998) Negotiating the Construction of Organisational Memories, In U.M. Borghoff and R. Pareschi (Ed.), Information Technology for Knowledge Management, pp. 55-78, Berlin: Springer.

BUCKLEY, C. AND VOORHEES, E. M. Retrieval system evaluation. In TREC: Experiment and Evaluation in Information Retrieval. MIT Press, Cambridge, MA, Chapter 3, 53–75. 2005.

Carroll, Ed. Estimating Software Based on Use Case Points. OOPSLA'05, San Diego, California, USA. 2005.

Chalé-Góngora, G., Llorens, J.. Gallego, E. Your Wish, My Command – Speeding up Projects in the Transportation Industry Using Ontologies. INCOSE International Symposium. Session 7 Track 5: Transportation. (2017).

Chitnis M., Tiwari P., Ananthamurthy L. UML Tools. 2003. http://www.developer.com/article.php/1593811

Davenport T. and Prusak L. Working knowledge: how organizations manage what they know. Boston Massachusetts: Harvard Business School Press, 1998

Demian, P., Fruchter, R. CoMem: Knowledge reuse from a corporate memory. Stanford University Thesis. 2001.

Dretske, Fred I. (1981). Knowledge and the Flow of Information. Cambridge: The MIT Press/Bradford. Books.

E.-A. Karlsson: Software Reuse – A Holistic Approach. John Willey and Sons. 1995.

Esteban J. Survey of Model-Based Systems Engineering (MBSE) Methodologies. ModelBased Systems Engineering (MBSE) Initiative International Council on Systems Engineering (INCOSE). INCOSE-TD-2007-003-01. 2008

Exman, I., Llorens, J., Fraga, A. SKYWare: The Unavoidable Convergence of Software towards Runnable Knowledge. JUCS (2015).

Floridi, Luciano. Is Semantic Information Meaningful Data? (2005) in Philosophy and Phenomenological Research, 70(2), 351-370.

Fowler, J. UML Distilled: A Brief Guide to the Standard Object Modeling Language. Addison-Wesley Professional. 3rd edition. ISBN-13: 978-0321193681. 2003.

Fraga, Anabel. Universal Knowledge Reuse Methodology. PhD Dissertation in the Carlos III of Madrid University. 2010

Fraga, A. Quality of Requirements in the Industrial Environment. Software Knowledge Workshop. IC3K Conference. 2015.

Fraga, A. Systems Engineering in the Industrial Environment: Requirements Engineering Processes. Software Knowledge Workshop. IC3K Conference. 2016.

Fraga, A. Extraction of Patterns Using NLP: Genetic Deafness. SEKE 2017: 428-431

Frakes W., Terry C. (1996), Software Reuse and Reusability Metrics and Models, ACM Computing Survey, 28(2), June, 1996.

Frakes, W., Kang K. Software Reuse Research: Status and Future. IEEE Transactions on Software Engineering. Vol. 31. No. 7. July 2005.

Frakes, William; Prieto-Diaz, Ruben; Fox, Christopher. DARE: Domain analysis and reuse environment. Annals of Software Engineering 5 (1998) 125–141. 1998.

Gallego, E. et al. Requirements Quality Analysis: A Successful Case Study in the Industry. CSDM 2016: 187-201

Gettier, Edmund (1983). "Is Justified True Belief Knowledge?". Analysis , vol. 23, págs. 121-123.

Hill et al. 2002. Integration of Knowledge Organization Systems into Digital Library Architectures. ASIST SigCR - http://www.lub.lu.se/SEMKOS/docs/Hill_KOSpaper7-2-final.doc. Last visited on 15th of November of 2007.

Hofkirchner, Wolfgang (1999). Towards a Unified Theory of Information. The Merging of Second-Order Cybernetics and Semiotics into a Single and Comprehensive Information Science. In: 15e Congrès International de Cybernétique, Namur 1998, Namur, pp. 175-180.

Hummel, O., Janjic, W., Atkinson, C. Code Conjurer: Pulling Reusable Software out of Thin Air. IEEE Software. September/October 2008.

International Conference on Software Reuse (ICSR2008). http://sei.pku.edu.cn/conference/icsr2008 (2008)

International Conference on Software Reuse (ICSR2009). http://icsr11.isase.org (2009)

J. Llorens, J. Morato, G. Genova, "RSHP: An information representation model based on relationships." In: Ernesto Damiani, Lakhmi C. Jain, Mauro Madravio (Eds.), Soft Computing in Software Engineering (Studies in Fuzziness and Soft Computing Series, Vol. 159), Springer 2004, pp 221-253. Available for reviewers in ftp://www.ie.inf.uc3m.es/llorens/ICSR.zip

Janée G., Ikeda S., Hill L. ADL Thesaurus Protocol v1.0. 2002. http://www.alexandria.ucsb.edu/thesaurus/protocol
http://nkos.slis.kent.edu/2002workshop/janee.ppt Last visited on 15th of November of 2007.

John F. Sowa: Knowledge Representation: Logical, Philosophical, and Computational Foundations. Brooks/Cole: New York, 2000

Jorgenson, D.W.; Griliches, Z. (1967). The Explanation of Productivity Change. Review of Economic Studies 34(99): 249–283.

Knowledge Reuse Workshop 2008. http://www.kr.inf.uc3m.es/KREUSE2008.htm (Last visited on 15th of Dec of 2017).

Knowledge Reuse Workshop 2009. http://www.kr.inf.uc3m.es/KREUSE2009.htm (Last visited on 15th of Dec of 2017).

KOS: Knowledge Organisation Systems. http://www.db.dk/bh/lifeboat_ko/concepts/knowledge_organization_systems.htm (Last visited on 15th of November of 2017).

KRAFT: Knowledge Fusion from Distributed Databases and Knowledge Bases.
http://www.csd.abdn.ac.uk/research/kraft.html (2000)

Kuhn, O. and Abecker, A. (1997) Corporate Memories for Knowledge Management in Industrial Practice: Prospects and Challenges, Journal of Universal Computer Science, 3(8), pp.929-954.

Llorens, Juan; Fuentes, José M.; Prieto-Diaz, Rubén; Astudillo, Hernán. Incremental Software Reuse. International Conference of Software Reuse (ICSR2006). Torino, Italy. 2006.

Lynne, M. Toward a Theory of Knowledge Reuse: Types of Knowledge Reuse Situations and factors in Reuse Success. Journal of Management Information
Systems. Vol. 18. No. 1, Summer 2001, pp. 57- 94. (2001)

MAJCHRZAK, A., NEECE, O. E., COOPER, L. P. KNOWLEDGE REUSE FOR INNOVATION – THE MISSING FOCUS IN KNOWLEDGE MANAGEMENT: RESULTS OF A CASE ANALYSIS AT THE JET PROPULSION LABORATORY. Jet Propulsion Laboratory, California Institute of Technology. 2001.

Manning C., Raghavan, P., Schütze, H. Introduction to Information Retrieval, Cambridge University Press. 2008.

Martin, James N., Systems Engineering Guidebook: A Process for Developing Systems and Products, CRC Press, Inc.: Boca Raton, FL, 1996.

Merriam Webster Dictionary Online II. Universal Term Definition. http://www.merriam-webster.com/dictionary/universal (Last visited on 15th of January of 2014).

Moffat A., Zobel J. Rank-biased precision for measurement of retrieval effectiveness. ACM Trans. Inf. Syst., 27(1):1-27, 2008.

Morisio, M., Ezran, M., Tully, C. Success and Failure Factors in Software Reuse. IEEE Transactions on Software Engineering. Vol. 28. No. 4. April 2002.

Mäki, Eerikki. Exploring and Exploiting Knowledge. Research on Knowledge Processes in Knowledge-Intensive Organizations. Helsinki University of Technology, Department of

Industrial Engineering and Management, Laboratory of Work Psychology and Leadership. 2008.

Neighbors, J. M., An Assessment of Reuse Technology after Ten Years, 3 rd International Conference on Software Reuse, pp. 6- 13, November 1994.

Pérez-Montoro Gutiérrez, Mario (2001). "La información como fundamento cognitivo de una definición adecuada de conocimiento". Extremeño Placer, Ana (ed.) (2001). La representación y organización del conocimiento: metodologías, modelos y aplicaciones. Alcalá de Henares, págs. 79-87.

Pérez-Montoro Gutiérrez, Mario (2004). "Identificación y representación del conocimiento organizacional: la propuesta epistemológica clásica". Barcelona: IN3-UOC (Discussion Paper Series; DP04-01). 29 págs. <www.uoc.edu/in3/dt/20390/index.html>. (Last visited on September of 2004).

Pérez-Montoro Gutiérrez, Mario (2007). The Phenomenon of Information. Lanham (Maryland): Scarecrow Press.

Pérez-Montoro Gutiérrez, Mario y Campos Havidich, Manuel (2002). Representación y procesamiento del conocimiento. Barcelona: EdiUOC.

Pérez-Montoro, Mario. Floridi's Semantic Model. BITrum. http://glossarium.bitrum.unileon.es/Home/Conocimiento/knowledge [last visited on April 2012]

Poulin, J. S., Caruso, J. M., and Hancock, D. R. The business case for software reuse. IBM Syst. J. 32, 4, 567–594. 1993.

R. Davis, H. Shrobe, and P. Szolovits. What is a Knowledge Representation? AI Magazine, 14(1):17-33, 1993.

Ronald J. Brachman; What IS-A is and isn't. An Analysis of Taxonomic Links in Semantic Networks; IEEE Computer, 16 (10); October 1983.

Semantic Conceptions of Information. In E. N. Zahlta (ed.) Stanford Encyclopedia of Philisophy. Stanford: The Metaphysics Research Lab. [Online] <http://plato.stanford.edu/entries/information-semantic/> (Last visited on November 2015).

Sonia Sánchez-Cuadrado, Jorge Morato, Vicente Palacios-Madrid, Juan Llorens-Morillo y José Antonio Moreiro-González. De repente, ¿todos hablamos de ontologías? El profesional de la Información, vol. 16, núm. 6, Noviembre-diciembre 2007.

Sturgeon, S., Martin, G. G. F. y Crayling, A. C. (1998). "Epistemology". En Crayling, A. C. (ed.) (1998) Philosophy 1. Oxford: Oxford University Press. Capítulo 1, págs. 7-26.

Sumanth, D. (1979). Productivity Measurement and Evaluation Models for Manufacturing Companies. Illinois Institute of Technology. pp. 291.

Turpin, Andrew; Scholer, Falk. User performance versus precision measures for simple search tasks. Proceedings of the 9th Annual international ACM SIGIR Conference on Research and Development in Information Retrieval (Seattle, Washington, USA, August 06-11, 2006) (New York, NY: ACM): 11–18. doi:10.1145/1148170.1148176

UML Standard specification. http://www.omg.org/spec/UML/2.4/ (Last visited in April 2017).

**Anabel Fraga** is a professional in Computer Engineering (1999). Prior to engaging in academic work, she was involved in the industry such as UNIX / Windows Systems Administrator, Telecommunication Applications (SICAP, Comptel, Nokia), project management and consulting. She obtained her Master's Degree in Electronic Commerce and Networks (2004), as well as the Diploma of Advanced Studies (2006) at Carlos III University in Madrid. She obtained her Ph.D. in 2010 on the topic of Knowledge Reuse. She is a professor at the Carlos III University in Madrid, also member of ACM CSTA, OMG, IASA, and INCOSE. She is currently Treasurer of the

INFORMATION SYSTEMS FRONTIERS. 21:5-25. 2019
(Published online on 06 June 2018) The final publication is available at Springer via:
https://doi.org/10.1007/s10796-018-9862-7

Spanish Association of Systems Engineering (AEIS) and coordinates financial management activities of the Knowledge Reuse Research Group. She has several publications in the area of knowledge management and software reuse, as well as two patents in exploitation in related areas. She leads several research projects related to the construction of information systems and knowledge reuse based on semantic structures.

**Juan Llorens** is Professor at the Informatics Department of the Carlos III University of Madrid – Spain. He received his MS degree in Industrial Engineer from the ICAI Polytechnic School at the UPC University in Madrid in 1986, Spain, and his Ph.D. in Industrial Engineering and robotics at the Carlos III University of Madrid, Spain in 1996. In 1987 he started his own company dealing with Artificial Intelligence applied to Database systems, named Knowledge Engineering SL (KE). Dr. Llorens worked as a technical director in KE and as Marketing Director in Advanced Vision Technologies until 1992 when he joined the Carlos III University. Since 2003 he is Professor at the Carlos III University of Madrid. His main subject is Knowledge Reuse, which he teaches in the Informatics studies at the University. Dr. Llorens is the leader of the KR Group (Knowledge Reuse Group) within the University. In 1998 he was invited to the Högskolan på Åland (HÅ) (Åland, Finland). From 1998 to 2008 he split his educational activities between Madrid's University and the HÅ, where he taught different Software Engineering subjects. He is a member of INCOSE's (International Council on Systems Engineering) Requirements Working Group and he is the President of the Spanish INCOSE Chapter (AEIS).

**Gonzalo Génova** is Associate Professor of Software Engineering at the Computer Science and Engineering Department, Universidad Carlos III de Madrid. He has an MS degree in Telecommunications Engineering, an MS degree in Philosophy and a Ph.D. in Computer Science and Engineering. His main research subjects are models and modeling languages in software engineering, requirements engineering, and philosophy of information systems.