Deliverable 19.2

Application of X-Media Technology to the Public Use Case

Authors: Johannes Busse
Ontoprise,
busse@ontoprise.de

Aba-Sah Dadzie
The University of Sheffield,
a.dadzie@dcs.shef.ac.uk

Lei Xia
The University of Sheffield,
l.xia@dcs.shef.ac.uk

Authors: Spiros Nikolopoulos,
CERTH,
nikolo@iti.gr

Thomas Franz,
University of Koblenz-Landau,
franz@uni-koblenz.de

Christoph Ringelstein,
University of Koblenz-Landau,
cringel@uni-koblenz.de

Authors: Alberto Lavelli,
FBK-irst,
lavelli@fbk.eu

Work-package 19: Public Use Case

Type: Public Deliverable

Distribution: Consortium

Status: Final

Date: 31.08.2009

Deliverable Coordinator: Alberto Lavelli

Reviewers: Steve Fullerton

Area Coordinator: Marina Giordanino, CRF

Project Coordinator: Fabio Ciravegna, The University of Sheffield

EU Project Officer: Leonard Maqua

ABSTRACT

The deliverable reports the current status of the public use case and describes the X-Media technology applied in the use case.

TABLE OF CONTENT

1 Introduction

As already reported in Deliverable D19.1, the definition of a public use case was needed to be able to publicly demonstrate the X-Media technologies. This is necessary because, due to 'Non-disclosure agreements' (NDAs) signed between the consortium and industrial partners, the research partners are unable to publish or demonstrate their research results or technologies with the datasets provided by industrial partners.

The aim of the public use case is to provide a data set and an ontology that are useable for public demonstration and publication purposes outside the consortium. The use case has to be able to accommodate the initial objectives specified by the original use cases from industrial partners. In order to test the technologies developed within the consortium it is necessary to provide a dataset that is large scale and able to accommodate a number of different information extraction tasks (namely, named entity extraction, relation extraction, image classification/clustering object recognition). In addition, the data set needs to be easily understandable [D19.1] without specific expertise.

These requirements were brought together to become the definition of the Bike Brake Fade use case, which is similar in spirit to the Rolls-Royce (R-R) Issue Resolution (IR) use case.

During the Timmendorfer meeting it was decided to perform the following actions (in response to the reviewers' comments on the report of the work done in workpackage WP19 at the review meeting):

- Enlarge the ontology
- Find more images
- Extract terminology from the ontology and annotate documents with such terms (which means also producing the corresponding triples as well)
- Make the Demonstration based on the XMediaBox GUI available on the Koblenz SVN

The sections of the deliverable are organized in the following way: first the X-Media technologies suitable for integration in the public use case are briefly described, together with the adaptation needed. Then the XMediaBox GUI is presented both in

general and in the specific public use case, providing a number of snapshots highlighting the main functionalities.

2 X-Media Technologies

2.1 Area 1 Technologies

2.1.1 Semantic Email with Process Support

Email is used as a mode for keeping track of one's tasks and it is thus a core element of personal information management practices [Ducheneaut & Bellotti, 2001]. When multiple workers collaborate in teams, e.g. in the analysis of a failure of a bicycle brake, email is used to coordinate and communicate tasks, documents, explanations and more. Although email connects individuals, information and tasks in a number of modalities, currently the knowledge embedded in email has not been exploited by standard software. To exploit the potential of email, X-Media is experimenting with the concepts of semantic email developed in WP4, and methods on process support developed in workpackage WP1.

In the following, we detail on the integration of semantic email with technology on process support and illustrate the user benefits by means of the bike brake scenario. Before, we briefly introduce the contributing components, namely COSIMail and DIALOG, as well as their technical integration.

2.1.1.1 COSIMail: Semantic Email

We designed and implemented a semantic email client, COSIMail (Figure 1), that establishes linkage between emails, associated persons, and the files that have been attached to an email message. In addition COSIMail provides access to email attachments that have been stored on a file system, e.g. a shared network folder, or the file system of a local computer. COSIMail enables to access such stored files directly from the email client. As a counterpart for COSIMail, we built a file manager extension, COSIFile (see Figure 2) that exploits semantic email by providing additional document attributes such as sender, recipient, subject, for documents that have been exchanged by email. Furthermore, it features direct access to the email associated to a file. A special entry in the context menu of a file enables to open the email in COSIMail directly.

This combination of tools has been evaluated [D4.3; Franz et al., 2009] and significant efficiency and satisfaction improvements of users have been measured.

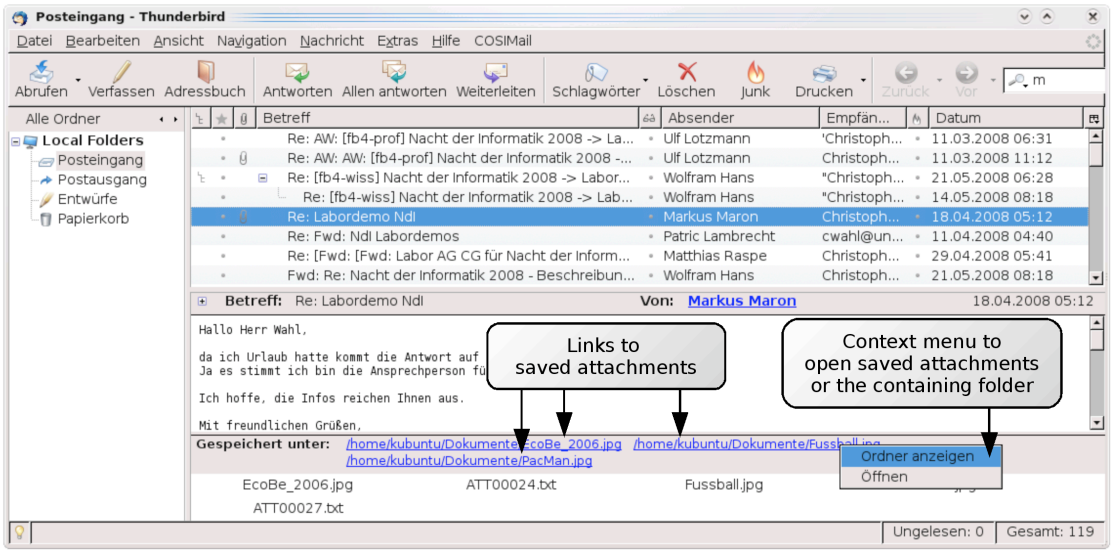


Figure 1. COSIMail

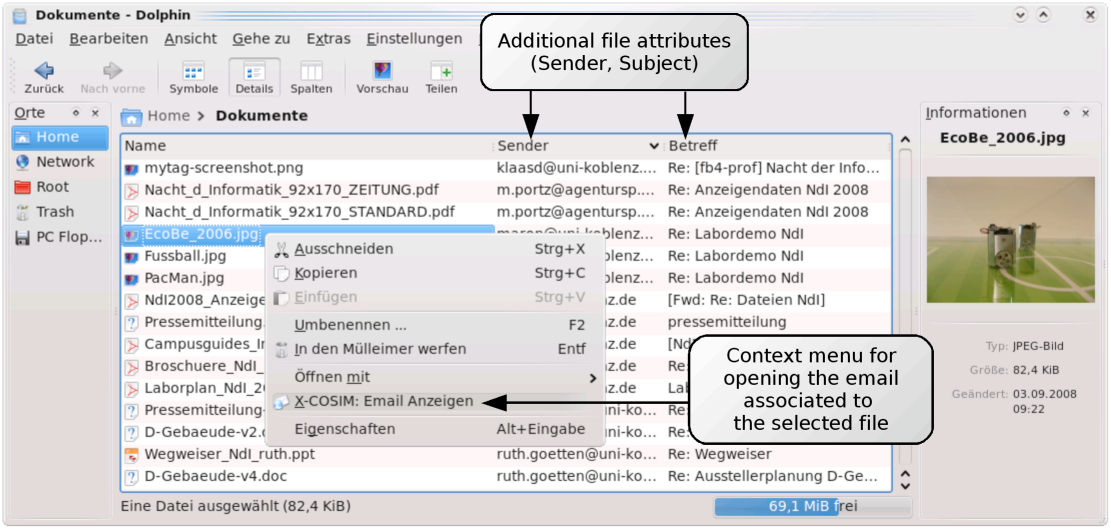


Figure 2. COSIFile

2.1.1.2 Process Support

We designed a formal model for process provenance, which we call DIALOG, that allows for modeling the distributed processing of data items (see Figure 3). A data item can be everything from a word to a document or file. The provenance information describes who changed when and where which data item. The formal model has been described in detail in X-Media Deliverable D1.8. In addition, we implemented the DIALOG as RDF-based logging mechanism that can be used by different kinds of

middleware, e.g. the JBoss message handler. The integration has been described in X-Media Deliverable D1.9.

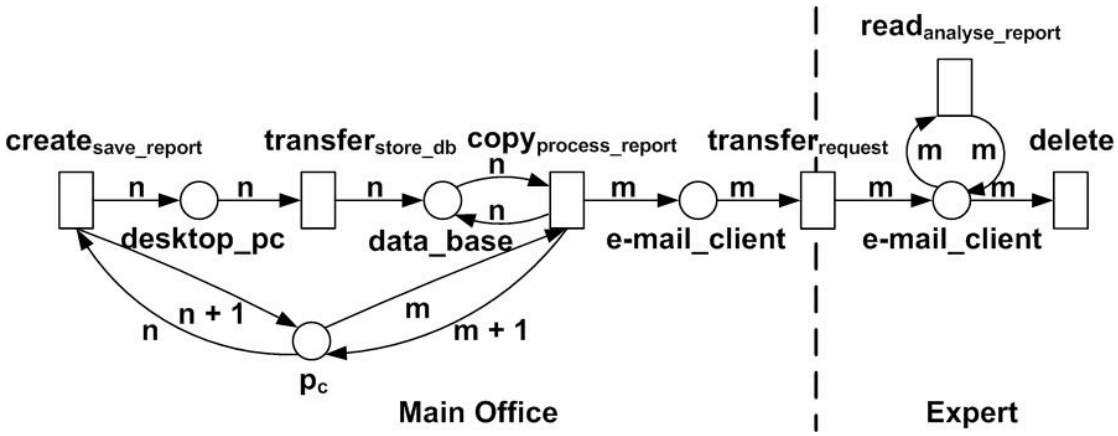


Figure 3. Formal Model of Process Provenance

2.1.1.3 Integration and Exploitation

As described above, the integration of COSIMail and COSIFile enables the connection of email attachments with local files. The integration with DIALOG allows for tracking the history of the processing of files across multiple desktops. For the user nothing changes; he still uses COSIMail and COSIFile to manage his files and emails. However, both tools provide provenance information by using the implementation of DIALOG. The integration of DIALOG also facilitates COSIMail to exchange the provenance information with other COSIMail clients. The provenance information can then be used to reconstruct the processing history of files, e.g. by utilizing LENA.

This combination of tools can then be utilized to keep track about actual bike-brake issue reports as well as to lookup the previous processes to gain insights about how reports of similar issues have been processed.

2.2 Area 2 Technologies

2.2.1 Workpackage WP5

2.2.1.1 Named Entity Recognition

Workpackage WP5 has developed a framework, called Runes, for the graphical representation of textual data and associated resources, in order to ease the feature extraction process. On top of Runes we have developed a graph-based, semi-supervised al-

gorithm. In order to perform entity extraction these algorithms are passed features extracted via Runes.

In the label propagation approach, known labels are used to propagate information through the graph in order to label all nodes. The goal is to learn a labelling function satisfying two constraints at the same time: 1) it should be close to the given labels on the labelled nodes, and 2) it should be smooth on the whole graph. The geometry of the data is thus captured by an empirical graph $G = (V; E)$ where nodes $V = 1,...,n$ represent the training data and edges E represent similarities between them, given by a weight or kernel matrix K such that K_{ij} is non-zero iff (i; j) $\in E$.

Starting with nodes 1, 2, …, l labelled with their known label (1 or -1) and nodes l +1, …, n labelled with 0, each node starts to propagate its label to its neighbours, and the process is repeated until convergence. An algorithm of this kind has been proposed by [Zhu & Ghahramani, 2002], which we reproduce here:

- Compute kernel matrix K
- Compute the diagonal degree matrix D by $D_{ii} \longleftarrow \sum_j K_{ij}$
- $P_j K_{ij}$
- Initialize $Y_0(y_0;::::;y_{1,}0,....,0)$
- Iterate
- 1. $Y^{(t+1)} \longleftarrow D^{-1} K Y^t$
- 2. $Y^{(t+1)} \longleftarrow Y_l$
- until convergence to $Y^{(\infty)}$
- Label point x_i by the sign of $y_i^{(\infty)}$

In the above, estimated labels on both labelled and unlabelled data are denoted by $Y = (Y_l, Y_u)$, and estimated labels may be allowed to differ from the actual labels. The algorithm is a particular form of the power iteration eigenvector algorithm, where Y is the stationary vector of the matrix K.

Also a rule-based document annotator is available. In this approach, rules that capture patterns in the text are manually written by the developers. Being the only non-machine learning-based approach provided by workpackage WP5, it is particularly suitable to be applied to text corpora which feature easily discernible structures and for which no annotations have been provided.

The rule-based extractor, called Saxon, is built upon the Runes framework. Saxon relies on the document being represented as a graph, with nodes representing document elements (tokens, sentences, etc.) and edges representing relationships between elements (this token occurs in this sentence, this token follows this token, etc.).

Saxon rules are defined as regular expressions detailing how to move between elements of the document. A rule has three main parts:

- a starting point;
- a regular expression describing how to move between sections of the document;
- a section detailing how the document should be updated if the rule matches.

The full flexibility of Saxon lies however in the ability to specify unrestricted Java code as the right hand side of a rule.

2.2.1.2 Relation Extraction

Our approach to semi-supervised relation extraction is simply the combination of the approaches mentioned in [D5.7] sections 2.2.1 and 2.1.2.. Concretely, we use the matrices pre-computed by the kernel functions K described in [D5.7] section 2.2.1 as the weight matrix W in the semi-supervised algorithms described in [D5.7] Section 2.1.2. The approach is thus similar to the graph labelling approach to relation extraction that can be found in [Chen et al., 2006]. The authors empirically showed that the label propagation algorithm achieves better performance than SVM when only very few labeled examples are available, and also outperforms bootstrapping methods for relation extraction task.

2.2.2 Workpackage WP6

2.2.2.1 Content-based image retrieval (CBIR)

The purpose of CBIR technology is to perform efficient content-based image retrieval on large-scale image datasets coming from highly heterogeneous environments, as in the case of industrial domains. The main two challenges that had to be addressed by this technology were to achieve meaningful image retrieval between visually heterogeneous data and at the same time adopt techniques that could efficiently cope with large-scale image repositories. In order to tackle the first requirement we have relied on various combinations of the MPEG-7 visual descriptors that capture different aspects of colour and texture [Manjunath et al., 2001]. With regards to the large scale requirement, multidimensional indexing structures based on R-trees were employed for achieving low response query time. However, the fact that the application of these structures can only be made feasible on a feature space with relative small dimensionality motivated the adoption of dimensionality reduction techniques such as Principal Component Analysis (PCA). PCA was applied on the original feature vectors in order to reduce their dimensionality to 20 dimensions. Subsequently, the reduced feature

vectors were indexed using the R-tree in order to provide for fast image search. Figure 4 demonstrates the process followed for indexing the images into an R-tree structure.

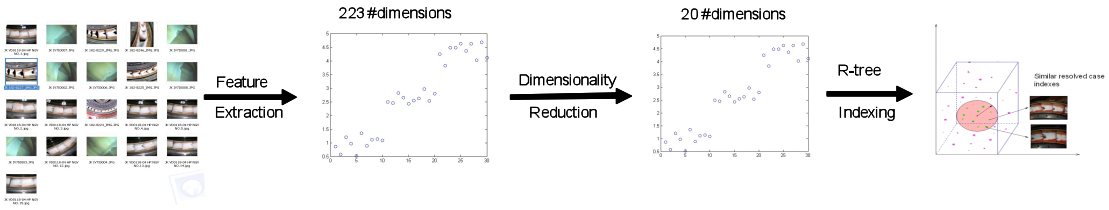


Figure 4. Technology for Content-Based Image Retrieval

2.2.2.2 Content-based image clustering (CBIC)

The aim of CBIC technology is to organize the image database according to visual content and, as much as possible, semantic content. Hence, the clustering process is performed in 3 steps:

1. Using a set of ground truth images (i.e. images which have been manually categorized), we build a clustering model that better fits the manual grouping. The idea is to recursively cluster the initial set of images using a single low level descriptor (from the MPEG-7 standard) at each stage of the clustering tree (see Figure 5). The clustering algorithm used was the dynamic clouds [Diday, 1971], which allows for more elongated and arbitrary shape than traditional k-means algorithm. The available ground truth categorization gives the actual number of clusters to obtain. Several different clustering trees can lead to this final number of clusters according to the set of descriptors.

2. When new images come, they are matched against the cluster model: at each stage of the tree, the image is assigned to the closest cluster, until it reaches a final cluster (see Figure 6).

3. After a period of time of exploiting the cluster model in phase 2, it is necessary to update it. The bigger set of images can be hence clustered, either using the same ground truth as the initial one, or using more examples for building the model if the user has manually validated new images into clusters.

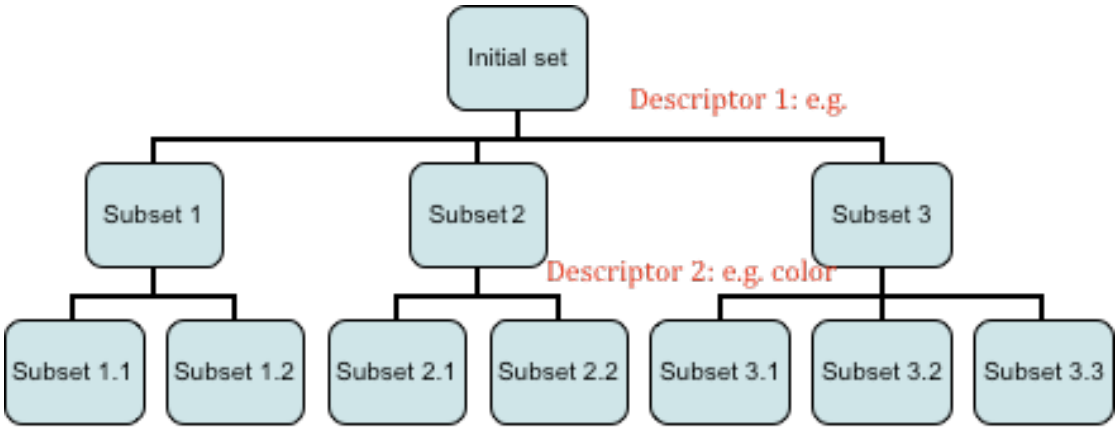


Figure 5. Building the Clustering Model

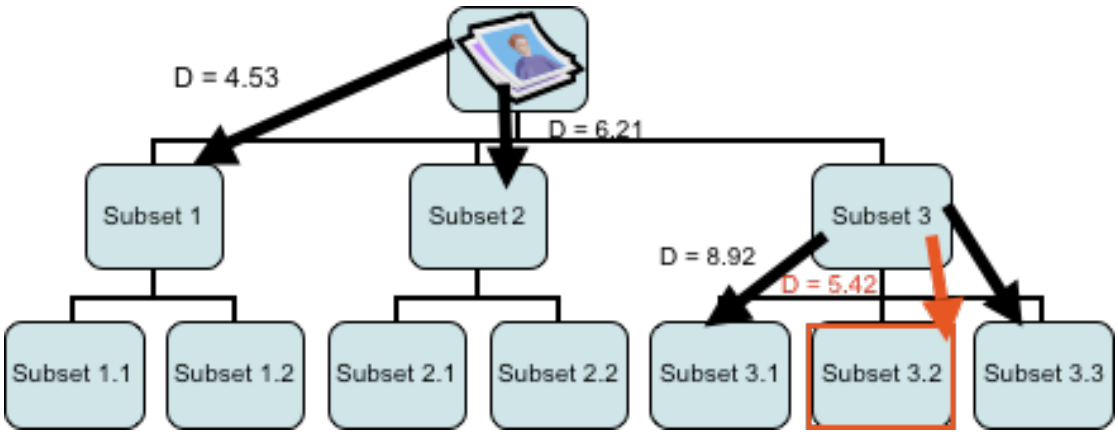


Figure 6. New Images are Submitted to the Model

For the purposes of the public use case we have used ~1,000 images depicting bike brakes that were collected from Flickr. All these images were processed using both CBIR and CBIC technologies that became part of the functionality of the X-Media box.

3 Adaptation to the Public Use Case

In D19.1 we have specified the methodology adopted to generate the dataset. The bike brake fade issues were selected as the topic of the public issue resolution use case. We also declared in D19.1 that the dataset is able to mimic the situations that appeared in the Rolls-Royce issue resolution use case where all around technologies were evaluated against it.

3.1 Resources

To support the new use case, the dataset has to meet the following conditions:

- Be large scale
- Be cross-media, e.g. text and images
- Contain issues specified in the 'Bike Brake Fade' ontology
- Contain named entities specified in the 'Bike Brake Fade' ontology

The data collection activity was started in May 2009, and more than 2GB of data was crawled from the internet. Although the data collection may seem a trivial task we have identified a number of challenges. In particular:

- Lack of specific bike brake issue resolution scenarios. Compare to Rolls-Royce and Fiat use cases, where large amounts of data are available from these companies' data archives. (Affects workpackages WP12 & WP13)
- Lack of expert knowledge/common sense of bike brake problem. Compare to other use cases, where engineers from respective companies provided us with their unique insights from their perspectives of how and what knowledge need to be targeted. (Affects workpackage WP3)
- Although there is a significant amount of bike brake images (querying GoogleImages for "bike brake disc" yields more than 200,000 images), textual document content is very limited. We have also discovered that there was not much interest from the public (bike users) where a failure of bike brakes was documented.

3.2 Template

Despite there being significant amounts of web resources for us to explore, collected data needs to be carefully filtered (possibly manually) to match what our ontology is able to deal with. The data set needs to contain significant amounts of textual information for workpackage WP5 (Information Extraction from Text) and cross-media data (mixture of text and image) for workpackage WP8 (Information Extraction from Cross-Media resources) to make use of. For example, resources that contain information or content describing a mechanical event, bike brake design specification, product specification, part specification, testing report and engineer reviews …

Since X-Media considers tens thousands of documents as large-scale, manual creation of this amount of data is practically unfeasible. To overcome this difficulty, we have developed a template filling strategy, which can be composed in the following steps;

1. An automated agent crawls bike relevant content
2. Fill the template with the data (text, images) into each section of the template
3. Template filling

   3.1 Lists of named entities (mapped with the concepts defined in the domain ontology) are randomly injected into the template.

   3.2 Lists of relation pairs (mapped with the relations defined in the domain ontology) are randomly injected into the template.

   3.3 List of images are randomly injected into the template.

This methodology allowed us to simulate a noise process over the dataset. Concretely, given a noise ratio R , we can specify how much irrelevant content can appear in the document. In addition, since we knew where and the total amount of named entities and relations we injected into the dataset, we obtained our Gold Standard from the extracted dataset.

3.3 Ontology

3.3.1 Crafting the bike brake show case ontology

3.3.1.1 The annotation tool mm2flo

Analyzing the X-Media requirements with respect to ontology engineering and ontology population leads to the idea that engineering an ontology (with a focus on the schema part) and populating an ontology (with a focus on the A-box part) doesn't necessarily call for distinct tools. Ideally there should be no tool change required between T-Box definition and A-box population. The requirements towards a text annotation tool suggest that there is no clear borderline between annotation as ontology population (adding facts to the A-box) and annotation as terminology refinement (adding new concepts to the T-Box).

Additionally it would be very nice if the same tool that will be used for ontology implementation also could be used by domain experts in the early phases of a semantic project, i.e. within brainstorming processes and other informal methods of knowledge elicitation.

Consequently we fathomed how the text annotation tool (which was to be developed in workpackage WP4) could be used for ontology engineering and ontology documentation. As a result of these genuine research activities we prototyped the tool mm2flo, which revives Donald Knuth's concept of literal programming: Instead of authoring ontology code and documenting it we now author a properly annotated, domain documentation in order to gather automatically the respective ontology from it. (The guiding paradigm "literate programming" is described nicely in http://www.literateprogramming.com/index.html)

In our development we realized the KISS principle: keep it short, simple. We strived at supporting end users, focusing on widely used standard modelling patterns while slowing down feature creep which often leads to overly complex and error prone software. One important goal was to minimize tool related training effort, allowing for generating uniform F-logic, OWL and RDF modelling on syntactical level (while accepting slightly different semantics in the respective languages).

The key idea revives the literal programming approach of Knuth to a "literate ontology engineering" paradigm, which identifies text annotation, ontology authoring and text authoring. We integrate visualization, structuring and authoring by allowing for authoring of T-box (classes) and A-box (facts) within the same authoring and interaction paradigm.

This key idea is realized by the interaction principle "tree tagging". The simplest modelling approach of maintaining content within a tree of elements serves as a fundamental overall structure. It is the mindmap itself that maintains the structure of the text and its annotations informally, while icon annotations define semantics (and edge annotations define text structure) formally. Doing so results in an isomorphism between text structure, semantic annotation and RDF(S) graph.

To our best knowledge we believe that mm2flo implements a genuinely new and innovative modelling approach, which is definitely not covered by current major ontology engineering workbenches. In 2002 there was an OntoEdit tool called mindmap2ontology available, which cannot be compared with the current approach. While the aged tool mindmap2ontology interprets a mindmap being a concept tree and nothing else, mm2flo **annotates** a mindmap with icons (). By interpreting a given (icon tagged) node with respect to its ancestors we are able to express schema, facts, rules and queries within the same interaction paradigm. In fact it is this mindmap annotation methodology that defines a new powerful but nevertheless simple interaction paradigm. End users (who often already use mindmaps for their personal knowledge management) now can author semantic models directly with a common and lightweight technology. Thus the tool also addresses the knowledge management aspect of X-Media.

We have outlined this idea already in X-Media Deliverable D19.1 (tool acronym "mm2o"). However, discussing the results of the last year's formative qualitative evaluation let to significant changes of the interaction paradigm. The most relevant improvement concerns the visualization of semantic tagging itself. In the December 2008 version we made use of edge labels within the semantic tree. Today we tag the nodes by themselves with icons. This simple move showed to be very efficient in improving a fast comprehension of a semantic map. It also led to the development of a significantly faster and shorter algorithm -- which in turn also made documentation (and thus communication) of the tool's capabilities much simpler.

The August 2009 version of the tool is currently under evaluation with several research and commercial end users. First results suggest that the tool is very easy to use, while some customers suggest adding some more sophisticated modelling capabilities. A new version that incorporates the evaluation results in depth is under development.

We have decided to publish the mm2flo as a functional prototype under GPL (as opposed to LGPL or the Eclipse) licence. Because most of the development efforts went

into various cycles of defining, prototyping and evaluating annotation principles (i.e. working on the logic of annotation) it now should be pretty easy for commercial 3rd parties to refactor the code if they want to exploit the idea by themselves in the long term. We ourselves already have customers who are happy to act as early adopters willing to contribute to our formative evaluation.

3.3.1.2 The bicycle brake universe

3.3.1.2.1 Upper level structures

Functionally equivalent to owl:Thing we allocate the upper level class TheBike-BrakeUniverse. Direct subclasses are the class entity , i.e. all things which can be seen as "some thing" in a narrower sense (be it abstract or concrete), and the class association , i.e. the class of reified properties between entities (things in a narrower sense).

The class entity can be partitioned into the primaryTree, i.e. the tree of classes, which form the taxonomy we are interested in, (c.f. Section 3.3.1.2.1.1) and the class modifier, i.e. those classes which are used in the axioms which describe the classes within the primary tree in more detail. (c.f. Section Modifier)

The class association is a domain of relations like

- has_certainty (range fuzzyValue)
- stated_by_agent (range annotatingAgent)
- stated_in_information_object (range information-object)
- has_time_stamp

An association can be seen as a relationship within an ER-Diagram. Associations are reified n-ary (n > 2) relations. We represent them as classes and interpret their properties as named relation parameters (this is exactly the way how associations in the XML Topic Map (XTM) paradigm work.). Having the relations reified anyhow it's easy to attach provenance information to them. This allows us to implement reasoning with uncertainty.

3.3.1.2.1.1 Primary Tree

The class primaryTree is a domain of the very generic relation has_sub (range is also primaryTree), which has the subrelations

- has_subAssembly

- hasPart
- has_subSystem

We have two peculiar import classes within the primaryTree:

- the subclass part (parts are physical endurants. They are located in time and space. As a crude simplification we also assume that non-parts like holes or pivot points are parts in a broader sense; c.f. Section 3.3.1.2.2.1)
- and the subclass abstractState (c.f. Section Abstract states)

Other rather generic subclasses of primaryTree are:

- physical-realization
  - information-realization
    - media (sth. that realizes data, i.e. the file you can retrieve from an URI)
- non-agentive-social-object
  - information-object
    - digital-data
      - multimedia-data
        - audio-data
        - image-data
        - text-data
        - video-data
        - XMLData
- annotatingAgent
  - person (a natural or legal person, or a group)
  - algorithm
- User
  - J_Flynn_Fletcher
- TimeInterval with the relations:
  - containedInTimeInterval (range TimeInterval)
  - overlappingWithTimeInterval (range TimeInterval)

  and instances like

  - 2009-04-17 → containedInTimeInterval → April2009
  - April2009 → containedInTimeInterval → Spring2009
  - Spring2009 → containedInTimeInterval → year2009
  - May2009
  - year2009
- Location with the relation locatedIn (range Location)
  - Country England
  - Region South_Yorkshire
  - City Sheffield
  - Neighbourhood Nether_Edge
  - Street Kenwood_Park_Rd
  - PostCode S7_xxx

3.3.1.2.1.2 Modifier

Modifiers are classes that are used mainly in the axioms of the primary tree or as values of properties. They act as distinguishing features ("differentia specifica") describing the characteristics of the classes of the primary tree. But how to decide whether a concept belongs to the primary tree or the modifier group? This is mainly a social question. Idea: We consider a "brown dog" rather being a dog (which would be the primary tree concept) which is somewhat brown (which would be the modifier concept then) rather than a brown-thing which is somewhat doggish. Though modifier concepts also can form trees, modifier trees normally are less deep and less tangled than the branches of the branches of the primary tree.

The class modifier has several (more or less abstract) subclasses:

- dependencyType (like symptom_of, prevents, contributes_to or correlates_with)
- method
  - testingMethod
    - brakeTest
- quality
  - observableQuality
    - vibrationQuality like flutter, resonance or brake_squeak
    - handlingQuality like brake_fade, excessive_heat or brake_blockage
    - materialQuality like erosion, abrasion, burning, Inclusion_high_density, inclusion_low_density, pore, micro_pore, cracking or shrinkage
  - qualityStandard (a term used in th CRF ontology. Something like a norm or poliy.)
- region
  - judgement
    - fuzzyValueJudgement (like impressive, good, bad, high, acceptable, ok, medium, low, insufficient, too_high, too_low, to_be_rejected, inacceptable, alarming, no_issue, weak, strong, always or never)
      - normalized_fuzzy_value (like top_ten, best_thirty_percent, average or bottom_ten_percent)
    - Issue_sentencing_value (like rr:Defects_rejectable_sentence, rr:Defects_acceptable_sentence or rr:No_defects_sentence)
    - dependencyModality (like always, often, sometimes, possibly, byChance, presumably or never)
  - measurement
    - shape (e.g. round, long_and_straight or deep)
    - location
    - interval (e.g. extension0-10mm, extension20mm_plusminus_ten_percent or extension2000-5000Hz)

3.3.1.2.2 Domain level structures

3.3.1.2.2.1 Parts

The bike brake ontology as used in the WP19 demonstration use case focuses on parts of a bike brake.

All these instances of part are connected to other parts

- either by means of the relation skos:narrower (range part)
- or the relation hasPart (range part)

One of its most important instances w.r.t. our usecase are i.e. bike, which in turn hasPart wheel | fork | seat | light | tube | drive_system | gear_system | brake_system | headset Parts contained in the ontology (as of July 2009) are:

- wheel hasPart rim | hub | tire
- fork
- seat skos:narrower saddle | seat_post
- light
- tube skos:narrower down_tube
- drive_system hasPart chainguard | pedal | chain | bottom_bracket | chain_ring
- gear_system skos:narrower front_gear_system | back_gear_system | gear_lever | derailleur | gear hasPart gear cog
- headset
- brake_system hasPart brake skos:narrower
    - front_brake
    - back_brake
    - drum_brake skos:narrower coaster_brake skos:narrower foot_brake skos:narrower
        - torpedo_brake
        - back_pedal_brake
    - caliper_brake skos:narrower
        - disk_brake uses_part disc_brake_rotor
        - rim_brake
            - hasPart brake_arm hasPart brake_shoe hasPart friction_pad and brake_pad
            - skos:narrower cantilever_brake skos:narrower direct_pull_cantilever_brake skos:narrower v_brake

brake hasPart

- brake_cable hasPart
    - housing skos:narrower cable_housing skos:narrower casing
    - wire skos:narrower inner_wire

        - cable →hasPart cable_strand | →skos:narrower bowden_cable
    - brake_lever →skos:narrower brake_handle
    - anchor_bolt
    - adjusting_barrel
    - bearing_adjustment
    - bolt →skos:narrower centre_bolt | anchor_bolt | holding_bolt
    - slack
    - nut →skos:narrower mounting_nut
    - cable_casing

brake_actuation_system

- side_pull_brake
- center_pull_brake
- dual_pivot_brake
- single_pivot_brake →skos:narrower Single-pivot_side-pull_brake_actuation | Dual-pivot_side-pull_brake_actuation
- mechanic_brake
- hydraulic_brake →hasPart hydraulic_fluid piston master_cylinder fluid_reservoir

3.3.1.2.2.2 Abstract states

We use the concept of an abstract state for "cognitive objects" like brake wobble insufficient brake pressure etc. One of the most important relationships between abstract states is causation.

Thus the class abstractState is domain of the relations: causes (with range abstractState) and coincides_with (with range abstractState)

Instances of abstractState relations:

- causes (range abstractState)
- repair range abstractState)
- coincides_with (range abstractState)

Abstract states describe the problems of bike brakes, like

- heated_rim →causes brake_fade | rim_warpage
- rim_warpage →causes brake_wobble
- brake_wobble →causes front_wheel_wobble | →related brake_pulsation
- too_much_clearance_in_the_brake_arme_bearing →related brake_bearings_clearance | →causes brake_wobble
- a_lot_of_wet_and_muddy_conditions_in_the_past →causes weared_down_brake_pads | worn_rim
- weared_down_brake_pads →causes brake_wobble
- pronounced_rim_wobble →causes front_wheel_wobble | insufficient_brake_pressure

- insufficient_brake_pressure causes brake_pulsation
- brake_pulsation causes brake_noise
- brake_noise skos:narrower brake_chatter | brake_squeal | brake_judder skos:narrower hot_brake_judder and cold_brake_judder
- knocking_in_the_headset causes brake_wobble
- dragging brake pads repair brake_adjustment
- creaking_or_clicking_noise repair apply_oil_to_spokes_intersections

3.3.1.2.2.3 Associations (i.e. reified relations)

Subclasses of association are:

- distance (A reified distance relationship between parts. Could be useful for image recognition. We hope that distance measurements can be estimated roughly from image annotations.) relations: from_part (range part) | to_part (range part) | has_value (range intervall)
- sentencing (The judgement of an expert whether an abstractState is ok or not) relations: has_judge (range person) | concerns_abstractState (range abstractState) | has_judgement (range judgement) | has_qualityStandard (range qualityStandard)
- nominalization The most generic association: Whereas in all the other associations the name represents the reified relation, we here enter a second level of abstraction: The relation which is reified is itself the value of a parameter, here: the has_quality attribute. relations
  - has_subject (range assembly)
  - has_quality (range observableQuality)
  - has_value (range region)
  - actualState relations:
    - satisfies (range abstractState)
    - has_testingMethod (range method)
- Observation relations
  - who (range User)
  - when (range TimeInterval)
  - where (range Location)
  - obs123
    - who J_Flynn_Fletcher
    - when April2009
    - where Nether_Edge
- dependency .Allows for expressing simple if-then rules w.r.t. abstractStates: "A heated_rim and rim_warpage often contributes_to brake_fade".Problem solving algorithms may use this information for (simple) reasoning. relations:
  - has_antecedent (range abstractState)
  - has_consequent (range abstractState)
  - has_dependencyType (range dependencyType)
  - has_dependencyModality (range dependencyModality)

Instances of dependency might look like:

- d25
  - has_antecedent brake_wobble

    - has_consequent front_wheel_wobble
    - has_dependencyType causality
    - has_dependencyModality always
- d24
    - has_antecedent weared_down_brake_pads | too_much_clearance_in_the_brake_arme_bearing
    - has_consequent brake_wobble
    - has_dependencyType contributes_to
    - has_dependencyModality often
- d21
    - has_antecedent heated_rim
    - has_consequent rim_warpage | brake_fade
    - has_dependencyType contributes_to
    - has_dependencyModality often
- d22
    - has_antecedent a_lot_of_wet_and_muddy_conditions_in_the_past
    - has_consequent worn_rim | weared_down_brake_pads
    - has_dependencyType contributes_to
    - has_dependencyModality often
- d23
    - has_antecedent heated_rim
    - has_consequent rim_warpage | brake_fade
    - has_dependencyType contributes_to
    - has_dependencyModality often

Instances of actualState might look like:

- myBrakeProblem-2007-05-21 has_subject my_caliperBrake | has_quality brake_fade brake_wobble | has_judgement bad
- myBrakeAccident-2007-05-23 has_subject my_caliperBrake | has_quality brakeBlockage | has_judgement inacceptable
- myBrakeRepair-2007-05-31 has_subject my_caliperBrake | has_quality brake_is_working | has_judgement ok

3.3.1.3 Discussion

We have shown reasons why we have developed a new annotation tool: We pushed the idea of "annotation" to its end: To craft a fully-fledged annotation tool means in fact to develop a literate modelling paradigm.

We have copied a lengthy text passage (section “3.3.1.2” which enumerates the elements of the ontology in detail) directly into this deliverable. The primary reason is to demonstrate the ontology of the public use case. Another though no less important reason is to show how to demonstrate an ontology using the newly developed tool mm2flo. The whole text was exported from a freemind map by mm2flo automatically into html. No significant additional formatting was applied to the text afterwards, only punctuation had to be amended. Mirroring the aim of workpackage WP19 (demon-

strating selected results of X-Media with a public sand box ontology) the current deliverable D19.2 serves as a sand box for demonstrating a tool's capabilities. The current section itself serves as a demonstration of how an automatically generated export of a semantically annotated mindmap to a classical office document could look like.

A remark. In our argumentation we refer implicitly to many ideas, thoughts and knowledge that stem from a long tradition of research and practice. However, we did not make use of specific arguments or proofs we can ascribe to single and identifiable authors or publications: Thus from a scientific point of view there is nothing we have to cite. We hope that our discourse contribution will be valuable to the reader without having to point to affirmative third party voices.

3.4 Evaluation

In this section we describe the criteria for us to identify usability of our public use case.

3.4.1 Large Scale

The automated template filling has virtually no upper bound for how much data we could generate. The template can be easy replaced with new ones; hence any document type (for example, diagnostic report or event report) can be easily created. In addition named entity, relation, image, problem description are pluggable, therefore, we can generate variations of data by plug-in or randomly mixing these modules. All Knowledge Acquisition technologies specified in deliverables D5.7, D6.5, and D8.9 are required to provide the capability of handling large-scale dataset.

3.4.2 Named Entity

For the named entity extraction task, we specified a few named entities for each concept (e.g. for date we have generated random dates ranging between January, 2000 - December, 2008) we defined in the domain ontology. We randomly injected those named entities into the document, with adjustable noise rate 'R' to simulate the context surrounding them. The data generated in such a way can then be provided as input to the IE tools specified in deliverable D5.7 (Terminology recognizer (TR), Trex+Aleph+Rune, Label-propagation, SVM, Gazetteer, Saxon).

3.4.3 Relation Extraction

Similar to the named entity extraction task, we also prepared two pairs of relations for each relation we defined in the domain ontology, in which both entities are subsets of named entities defined in the domain ontology. In addition, the relation extraction method largely depends on analysing the pre, middle, and post context to predict a relation pair, hence, we artificially created these contexts to fulfil this requirement. Technologies defined in deliverable D5.7 for relation extraction are users of our dataset for performance evaluation.

3.4.4 Image

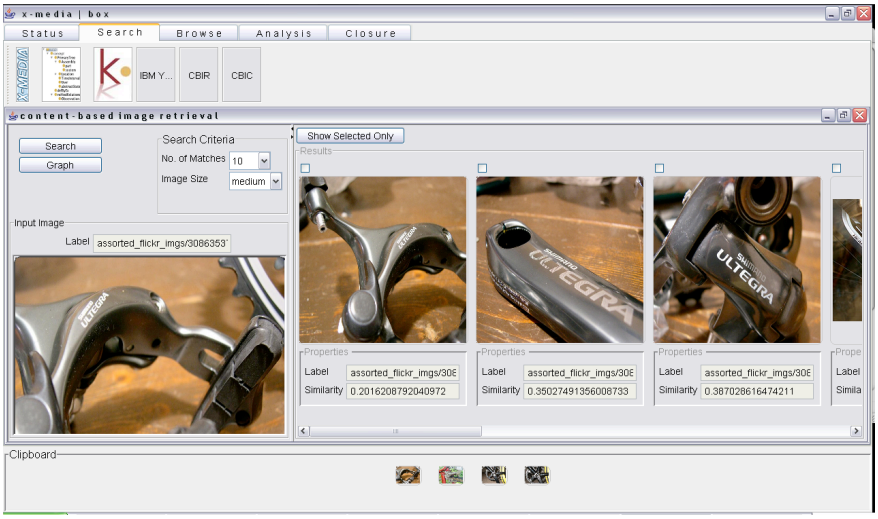
We injected bike brake images, downloaded from the web. Most of images are in high quality in human eyes and focused on bike brakes. We inject five images per document that are comparable to Rolls-Royce strip-reports. Semantic image clustering [D6.5], and image retrieval [D6.5], and object recogniser are the end users for our image dataset.

3.4.5 GUI & User Evaluation

X-Media uses the 'X-Media Box' to bridge the interaction between user and system (namely structured/unstructured data). Five tabs that support different stages in the Issue Resolution process, Status, Search, Browse, Analysis, Closure. The operability of the following three tables demonstrates the successfulness of our dataset.

Status: used to capture the Issue Definition trigger initial legacy corpora search; our dataset is able to provide the scenario to support this process.

Search: K-Search, a hybrid (ontology + keyword) search tool support both. It can perform semantic search and keyword search over both text and image, which have been indexed.



Our data support multi-media content, where text and images are included in each document.

4 XMediaBox

The current version of the XMediaBox is in use for the R-R Issue Resolution use case and the public Bicycle Brakes use case. It has been tested with XMKernel 1.0.1-dev, KnoFuss 0.1.4-dev, X-Media Summarizer 1.1.1-dev, R-R Issue Resolution 0.1-dev, K-Search, and the University of Sheffield Terminology Recogniser module TRv1.2b. Dependencies include Prefuse, SwingWorker, JAI_codec, JAI_core, JDOM, JENA, MySql Connector J, and log4j.

The GUI is composed of 5 tabs that support the following stages in the Issue Resolution process:

- Status
- Search
- Browse
- Analysis
- Closure

plus the Semantic Clipboard and the Ontology Browser.

4.1 Status Tab

The Status Tab is used to capture the Issue Definition (see Figure 7).

- trigger initial legacy corpora search
    - currently only K-Search but will be extended to “integrated” search

Process support tools for tracking the status of the investigation include

- Issue Summary Table
    - description of current issue with supporting metadata
    - instances of related issues used to calculate *disruption index*
- Investigation Tracking (Contrasts & Comparisons) Table

- thinking and analysis tools – free text in cells, each supported by evidence in formal documents and other knowledge objects
- will be kept in XMediaBox but no further development to be done

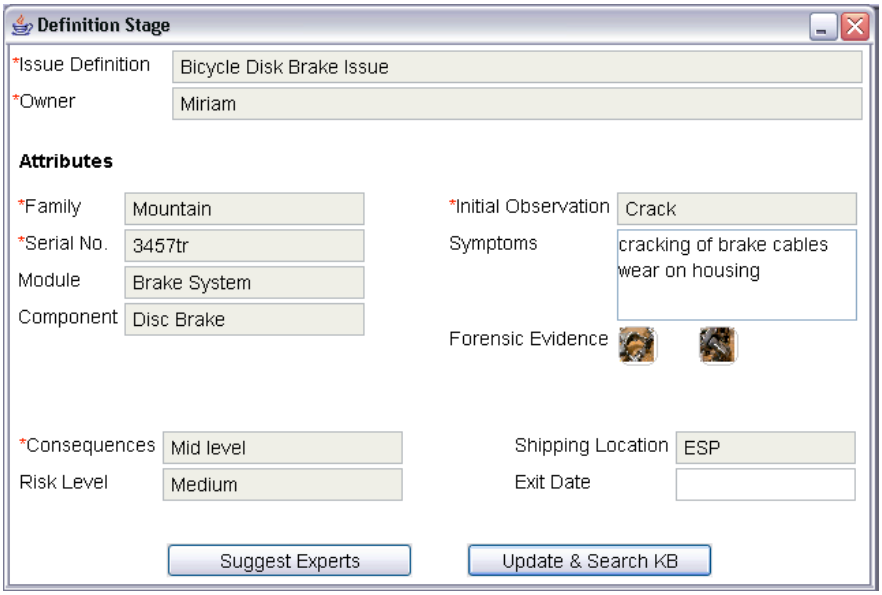


Figure 7. Issue Definition Form

4.1.1 Definition Form and K-Search trigger

It currently demonstrates the ability to use the K-Search API.

Users may correct Terminology Recognition (TR) output; results are saved and used to trigger the initial legacy corpora search (see Figure 8).

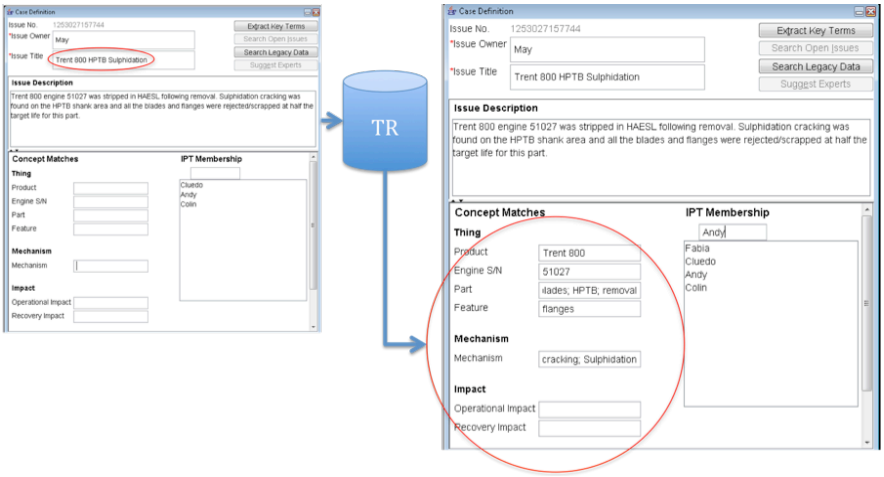


Figure 8. Issue Definition Form + Terminology Recognition

TR results are less accurate than for Rolls Royce key terms because the module runs on domain-specific terminology.

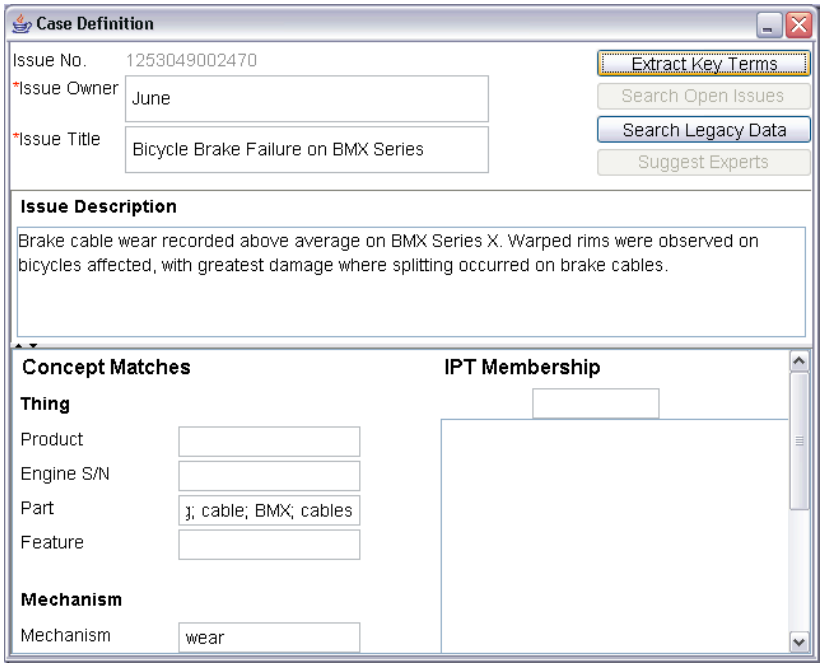


Figure 9. TR on Bike Brakes Issue

Legacy data search makes use of the K-Search API. A keyword-in-concept query is built by matching the terms to corresponding concepts in the ontology.

Once the search is triggered it checks if TR has been previously performed, otherwise it runs the module, then releases the primary thread and continues in the background, enabling further interactivity while waiting for the results.

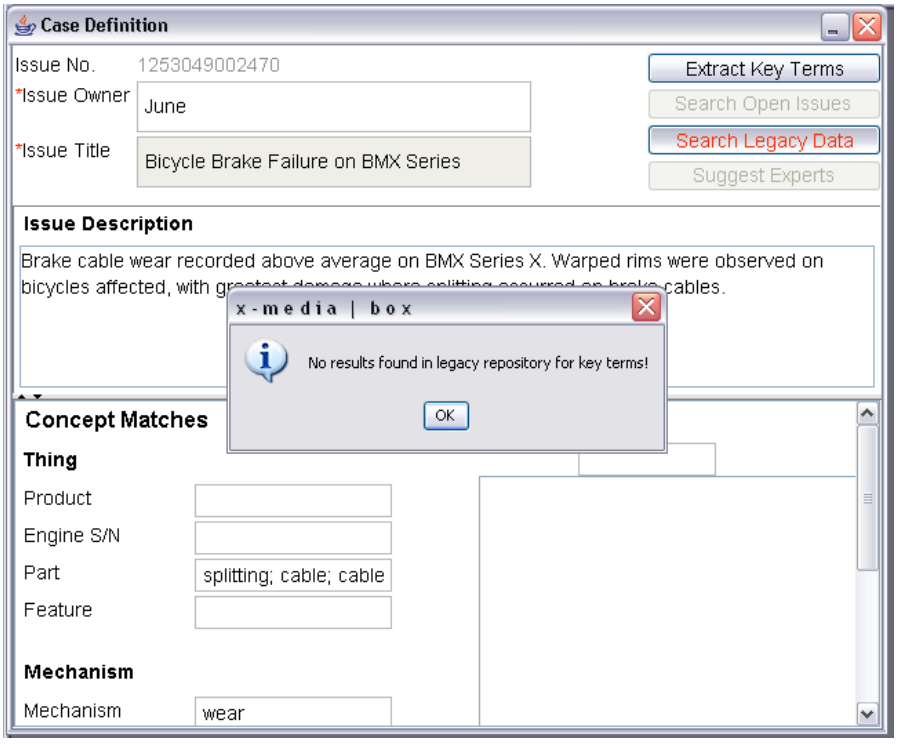


Figure 10. Initial Search on Legacy Data

The Issue Summary table (Figure 11) provides more detail about the issue being investigated, and allows references to (instances of) related events to be retrieved and supports the exploration of their relationship to the current issue.

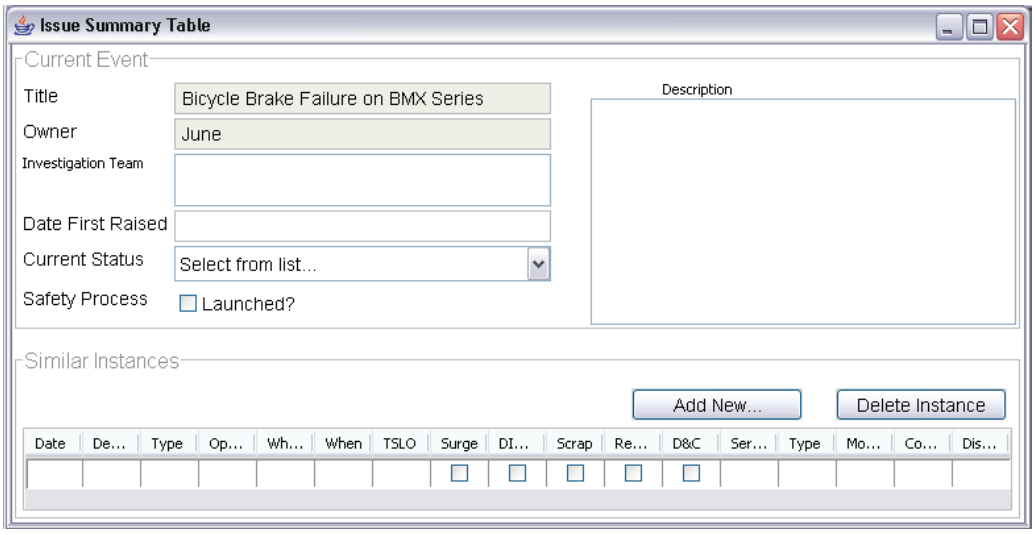


Figure 11. Issue Summary Table

4.2 Search Tab

- K-Search

- hybrid (ontology + keyword) search
  Note that K-Search does not directly do image search – it retrieves related images that have been (previously) annotated or where automated Knowledge Acquisition has found a caption that describes the image content.
- currently not directly integrated into framework
  - link to web version
  - use of K-Search API to perform background searches

- for IR – link to IBM Omnifind Yahoo (http://omnifind.ibm.yahoo.net)
  - aim is to allow comparison with other search modules
- Image Similarity Search
  - Content-Based Image Retrieval (CBIR)
  - Content-Based Image Clustering (CBIC)

4.2.1 CBIR & CBIC Modules

Snapshots of the two modules are shown in Figure 12, Figure 13, Figure 14 below.

- pre-indexed images and cross-media files
- input may be from
  - the pre-indexed files - retrievable by browsing the semantic repository (Browse Tab)
  - a new image or cross-media document imported into system via Investigation Workbook (Browse Tab) – this will trigger the indexers in import
- returns matching standalone images or documents containing matching images

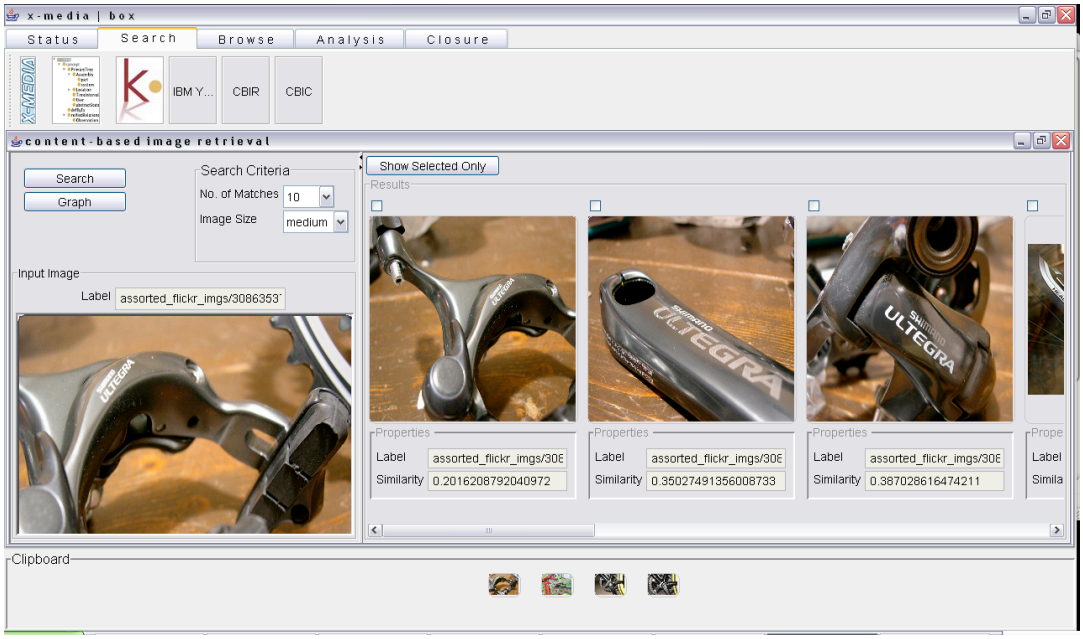


Figure 12. Content Based Image Retrieval

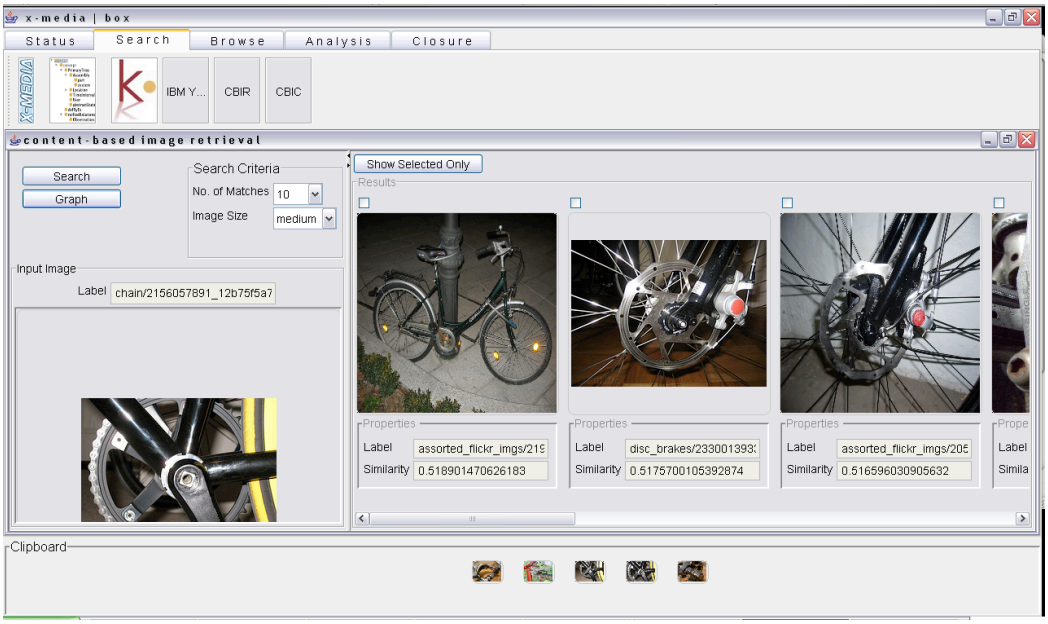


Figure 13. Content Based Image Retrieval

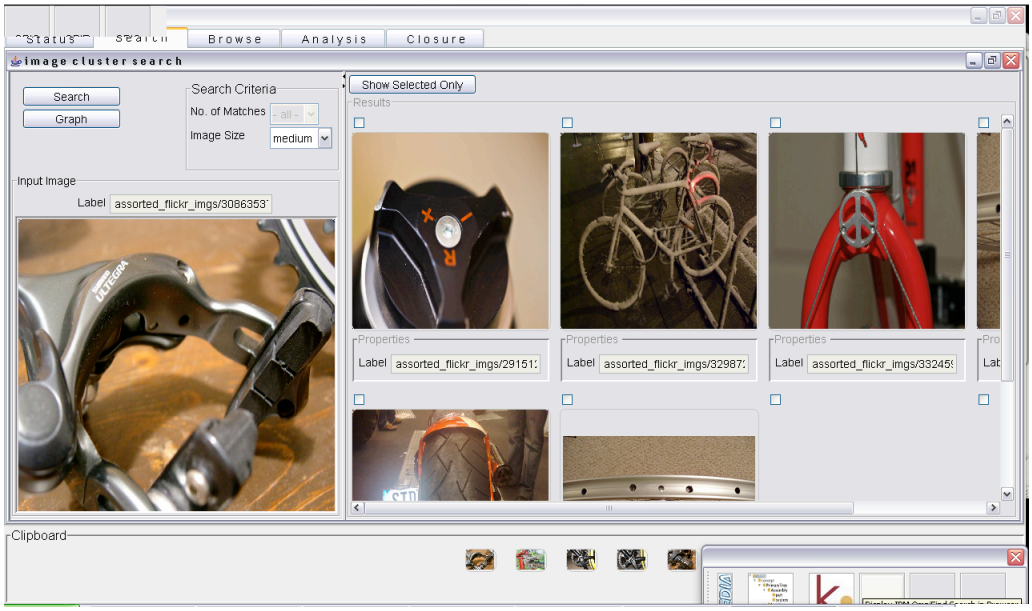


Figure 14. Image Cluster Search

4.3 Browse Tab

The three main elements are the following:

- Investigation Workbook (formerly known as the Sandbox)
    - contains references to all data and knowledge objects in use within the user's workspace
    - displays selected metadata for each
- Semantic Repository Browser
    - X-Media semantic repository browsing by XMSource
- CognIT XMSummarizer (Figure 17 and Figure 18)
    - The CognIT Summarizer analyses documents and extracts the core concepts and named entities as well as associations between them, thus establishing a lightweight concept graph.

4.3.1 Semantic Repository Browser

The Semantic Repository Browser lists all defined XMSources available for the current session type.

- allows the user to browse the X-Media semantic repository by source type
    - thumbnails are drawn for image files
    - all other document types are visualised using a representative icon
- Drag and Drop (DnD) may be used to import any document into the user's workspace.

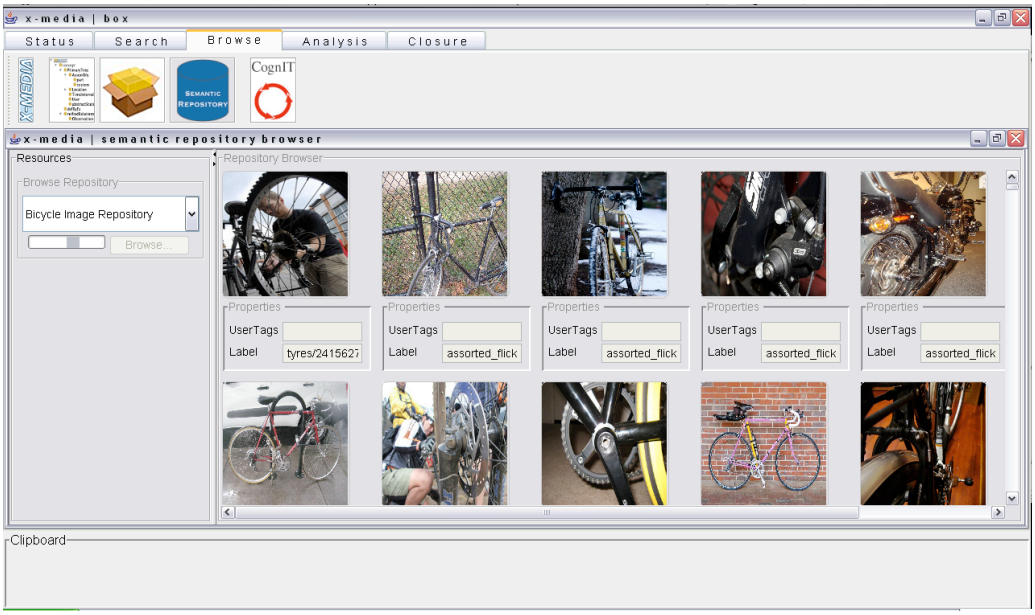


Figure 15. Semantic Repository Browser

It is possible to browse through pre-defined X-Media kernel sources. The data load uses a background thread, enabling users to switch to other modules while the data is retrieved. The action may be cancelled at any time.

To save a reference to the user's workspace a thumbnail is dragged into the clipboard at the bottom of the window.

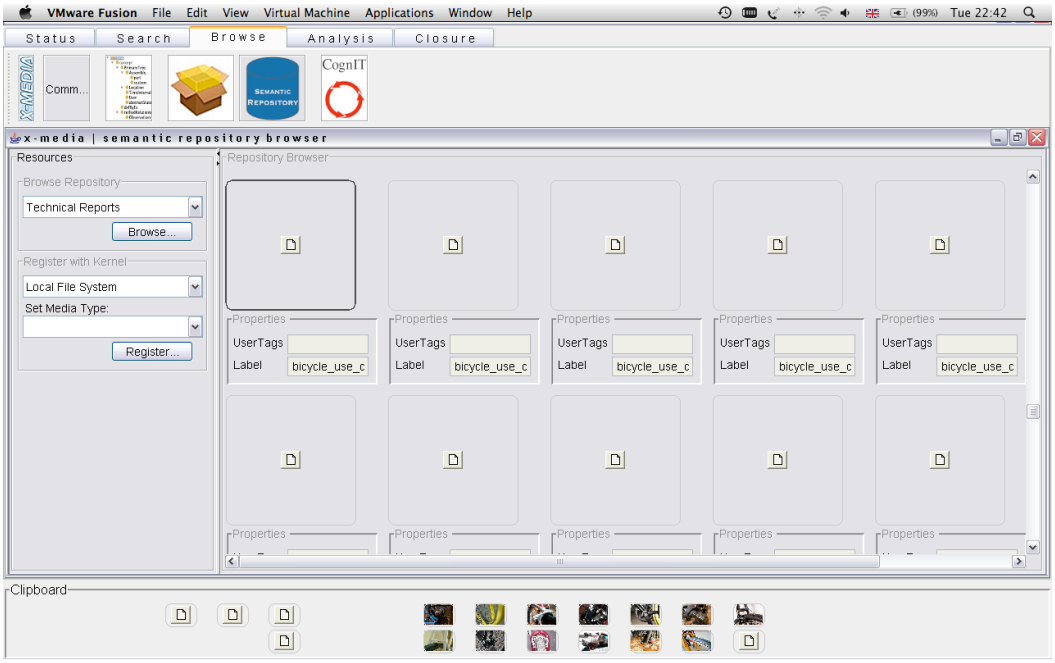


Figure 16. Semantic Repository Browser

Data references are imported into the user's workspace using Drag and Drop from the browser window.

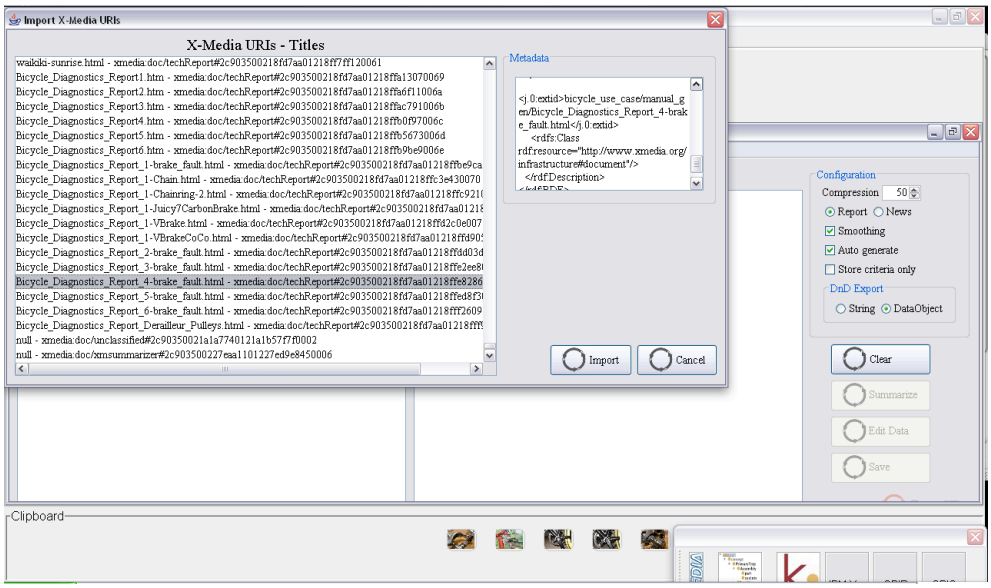


Figure 17. CognIT Summarizer

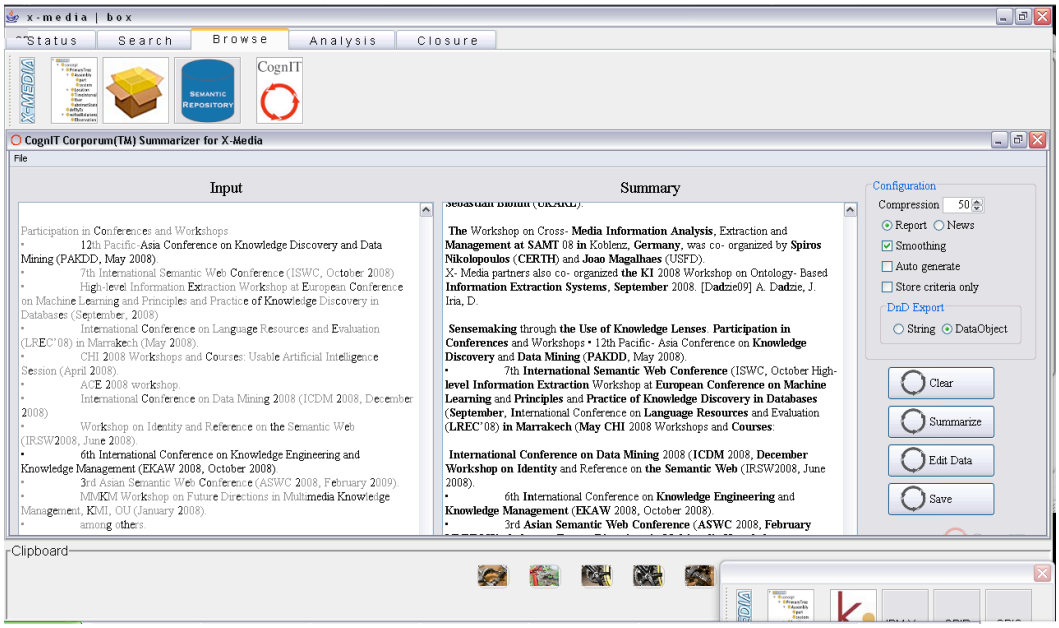


Figure 18. CognIT Summarizer

4.4 Analysis Tab

K-Views

- analysis tree
    - hierarchical hypothesis investigation based on the domain ontology
    - graph construction is guided by the (user-selected) ontology
    - image export has been completed – the whole graph draws, also the entire root cause analysis or the complete investigation may be exported to XML and formatted HTML with all supporting evidence (see Figure 19 and Figure 20 below)
    - TBD: allow changing of guiding ontology during tree construction

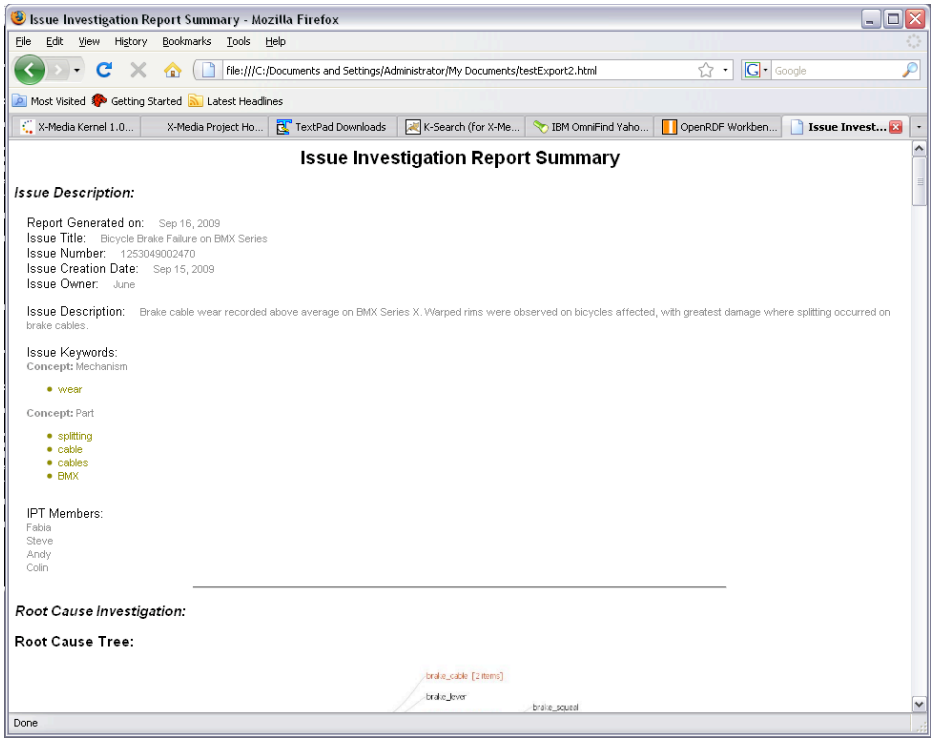


Figure 19. Issue Investigation Report Summary

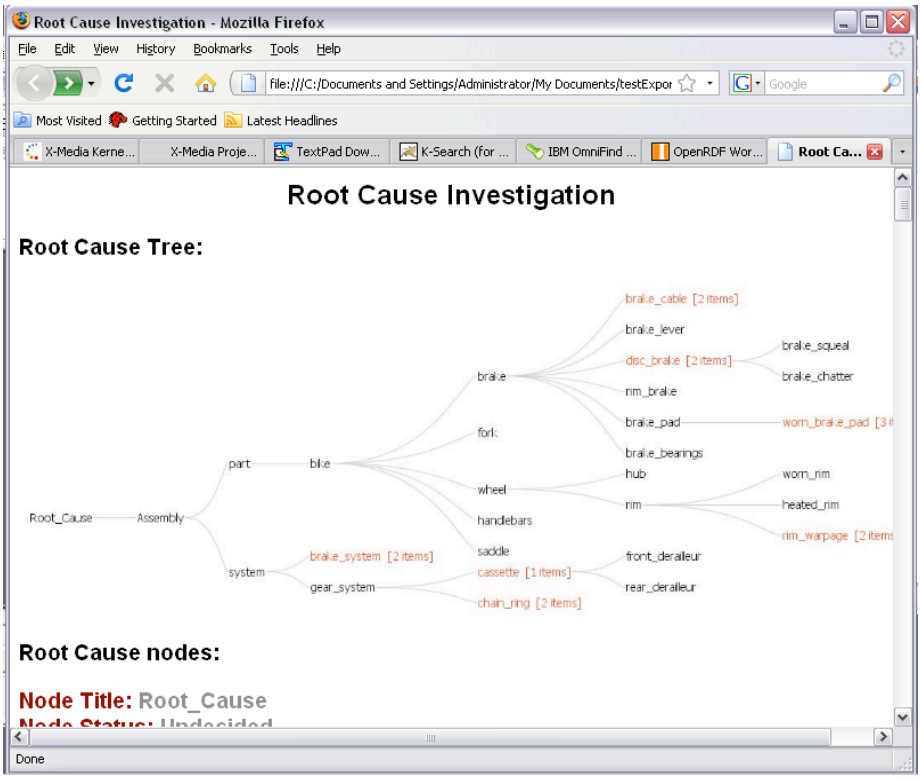


Figure 20. Root Cause Investigation Report

- knowledge cloud
    - support for browsing the semantic repository based on the domain ontology structure
    - co-ordinated perspective on analysis tree using a semantic network
        - nodes in the tree are superimposed on the visualisation of the ontology and repository
            - thick border indicates a node is both in the analysis tree and defined as a concept in the ontology
            - a broken border indicates a node is in the analysis tree but represents a new concept defined by the user

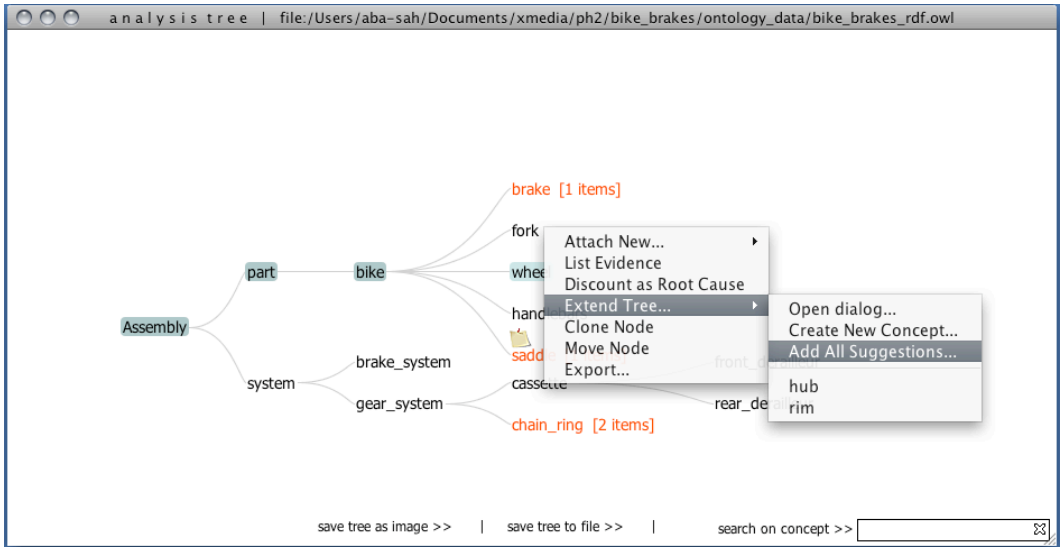


Figure 21. K-View Analysis Tree

A look-up displays, where found, concepts and/or instances related to that selected. A root node, "Root_Cause", is automatically pre-pended to the tree, functionality then provided for extending the tree, either according to the ontology structure and/or by including user-specified terms and relations, using the context menus and the evidence viewer attached to the window.

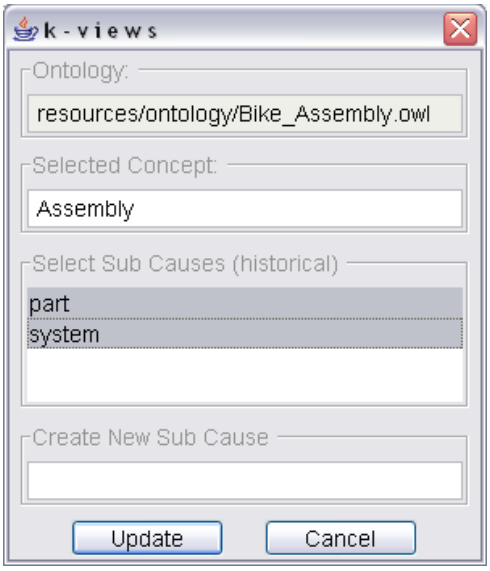


Figure 22.

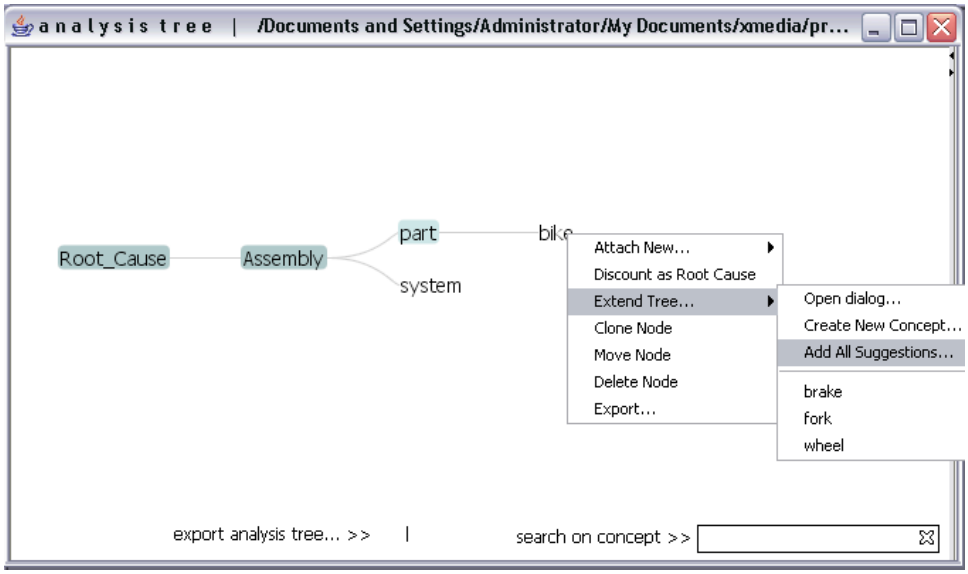


Figure 23.

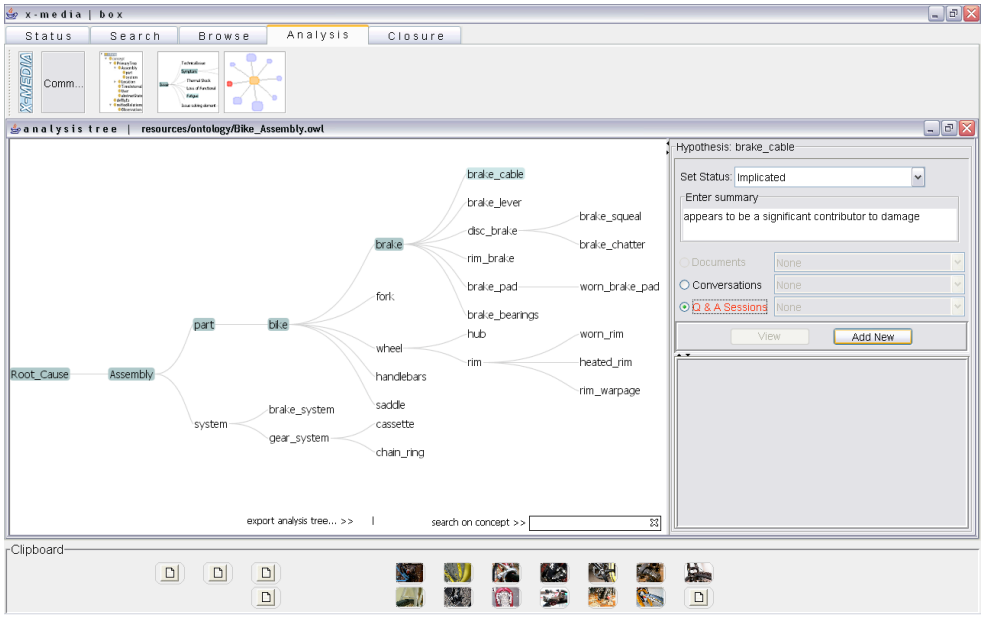


Figure 24. Evidence Viewers

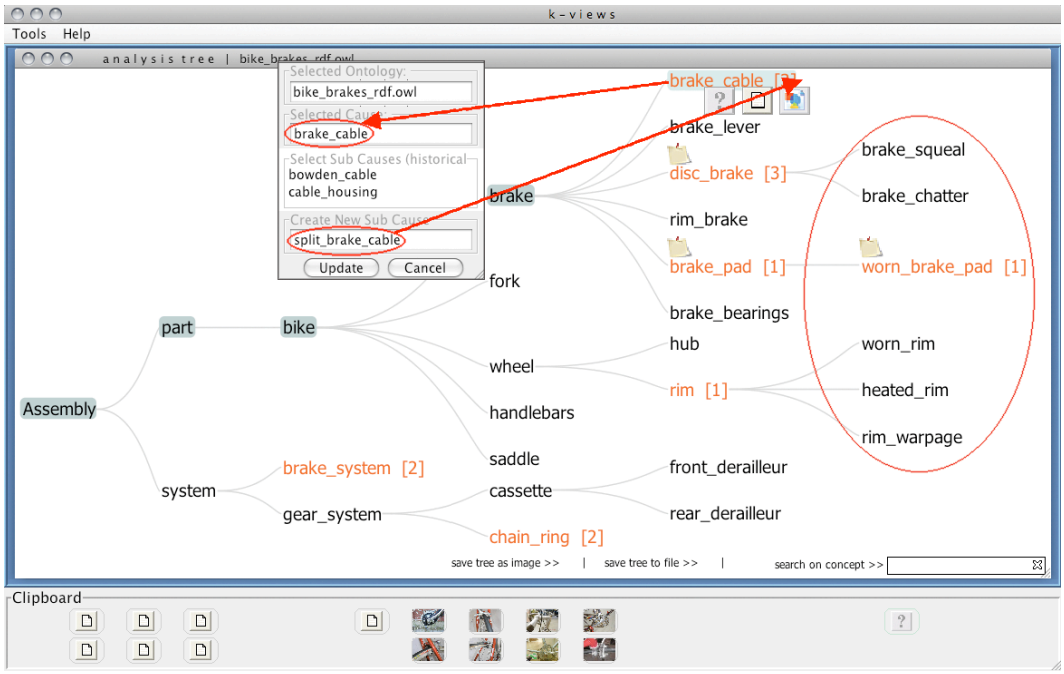


Figure 25. K-View Analysis Tree

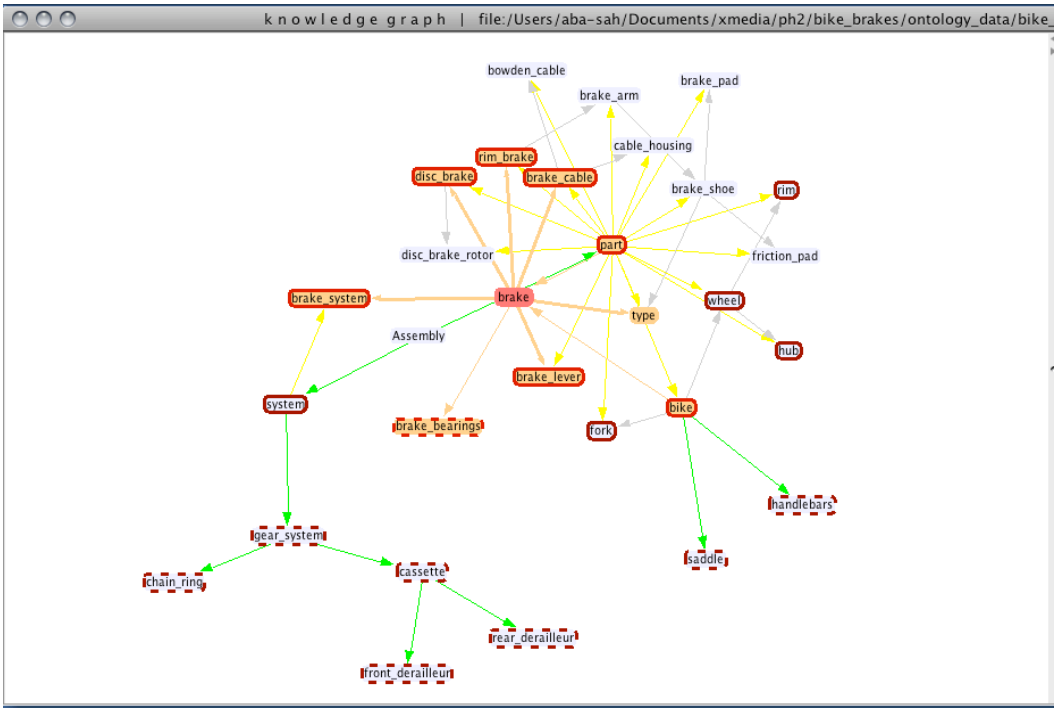


Figure 26. K-View Knowledge Cloud

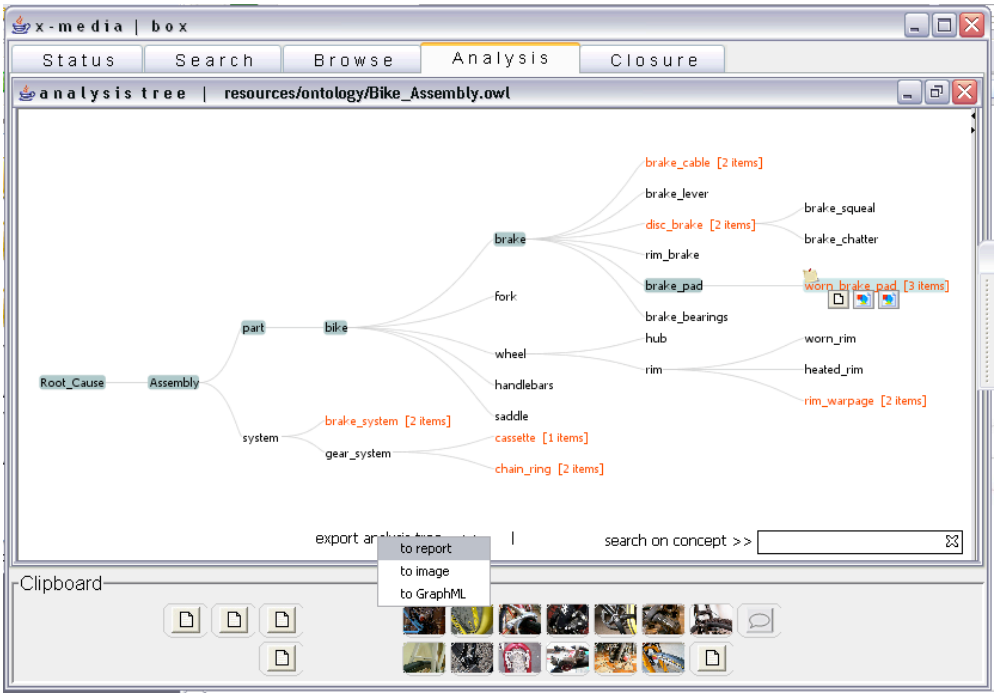


Figure 27. K-View Analysis Tree

Figure 24 shows the evidence viewer that works with the analysis tree. It is now part of a split pane connected to the Analysis Tree window, and provides support for setting each node's status, e.g., discounted, implicated, and for adding a brief summary of the hypothesis each node represents.

4.5 Closure Tab

The possibility of exporting the Issue Investigation in report form (formatted HTML) is available from this tab (see Figure 29).

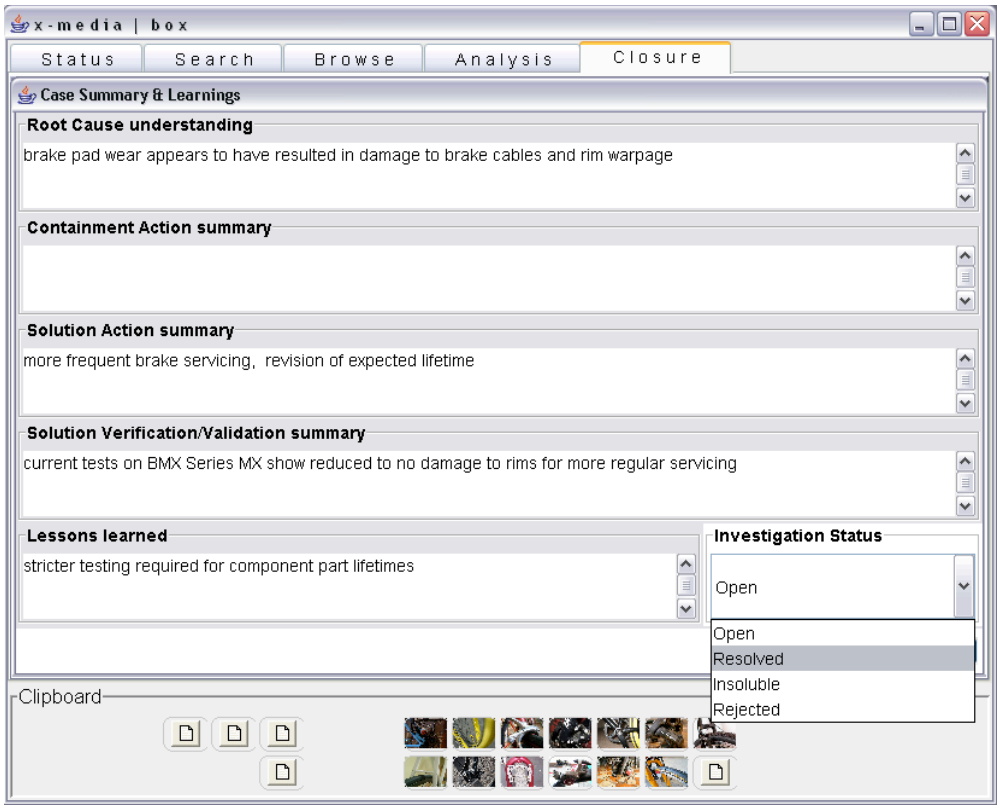


Figure 28. Closure Tab

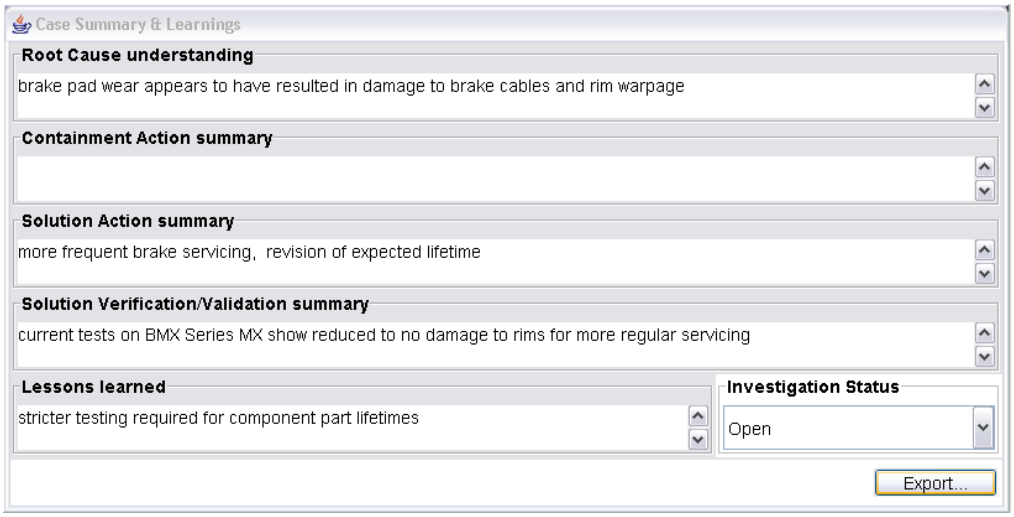


Figure 29. Exporting the Issue Investigation in report form

4.6 Ontology Browser

The Ontology Browser (Figure 30) allows the browsing of the structure of pre-specified ontologies (*Perspectives* on data or knowledge).

Concepts in any of the ontologies loaded may be used to tag data and knowledge objects within the system.

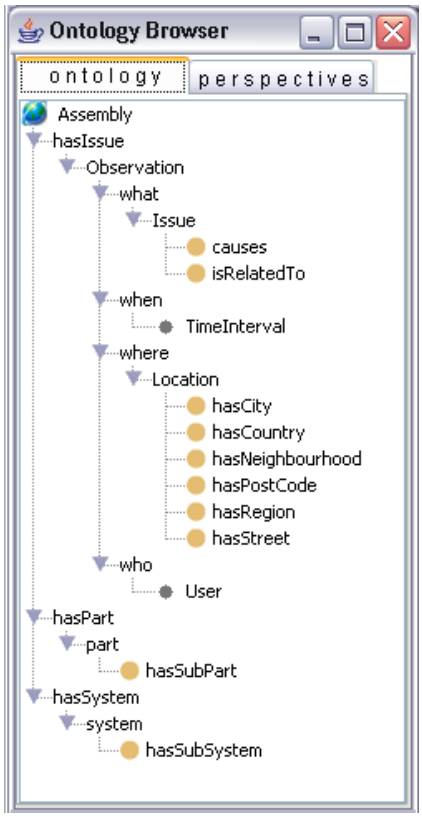


Figure 30. Ontology Browser

An ontology browser supports browsing through a set of pre-specified ontologies (in this case mapped to those currently loaded into K-Search relevant to the current session type).

Selecting a concept from the tree allows users to perform simple tagging of XMDataObjects using the Drag and Drop (DnD) feature. The text entered by the user is saved as metadata to be reused in classification/categorisation and keyword-in-context queries for retrieving similar items.

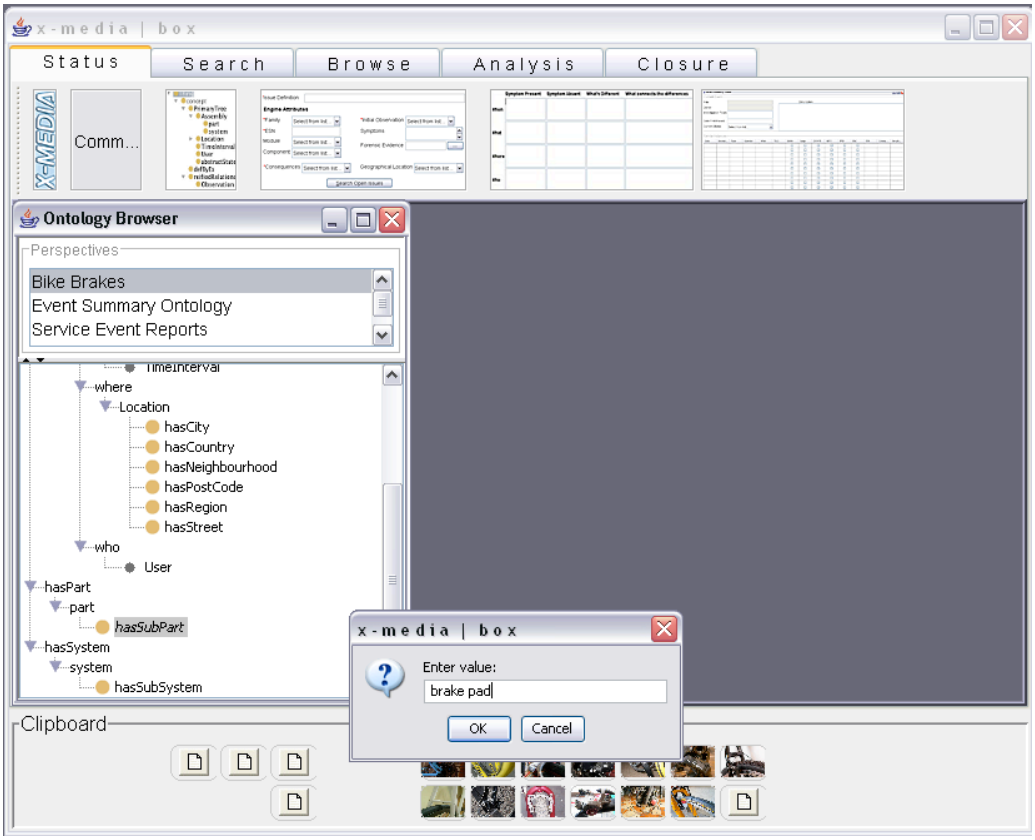


Figure 31. Tagging by Drag and Drop (DnD)

4.7 Semantic Clipboard

The Semantic Clipboard provides a filtered, temporary store displaying selected items in the "workbook" (in the Browse tab). It is shared across the application, supports data and knowledge exchange between modules

- using drag and drop (DnD) functionality
- context menu used to provide more information about each object and further actions

XMDataObjects are references to legacy data or the conversation and Q&A sessions created within the framework). These may be retrieved from the *Investigation Workbook* in the Browse Tab and reused in different modules by dragging the visual representation between windows.

The *Semantic Clipboard* provides a shareable, always visible, temporary holder for XMDataObjects in current use.

The tag attached to the XMDataObject with the focus is stored as metadata, and is easily retrieved using the context menu attached to each XMDataObject.

Figure 32. Semantic Clipboard & XMDataObjects

4.8 Offline Indexing

It is currently triggered on single file import of image, text or cross-media files, if one of these settings is selected on import in the Investigation Workbook (Browse Tab).

Batch processing of imported documents is now available from the Browse Tab, using the X-Media Semantic Repository Browser.

It will make use of a set of bins/facts for visual categorisation to ensure correct XMSources used and appropriate indexers triggered.

4.9 Capturing the Results of User's Actions

Different possibilities of capturing the results of users' actions are available:

- user actions captured to a SystemLog
    - writes out a timestamped, text file on normal exit
- session saved to serialised Java object
    - using XMUsername and session type (use case type)
    - allows reloading of session
    - on login the user continues with a previously saved session (stored with user name and session type – so may not retrieve another user's session) or creates a new session. The session label is the issue investigation title in the definition form in the Status Tab. The user will be

prompted to set this on normal exit if not already done – the only restriction is that each title for each user must be unique.

- support searching across investigations

4.10 Q&A Session Record

Q&A Session Records (actions in exported reports) – also mention conversation records (discussions/minutes in exported reports) – same but do not include requests that need to be completed.

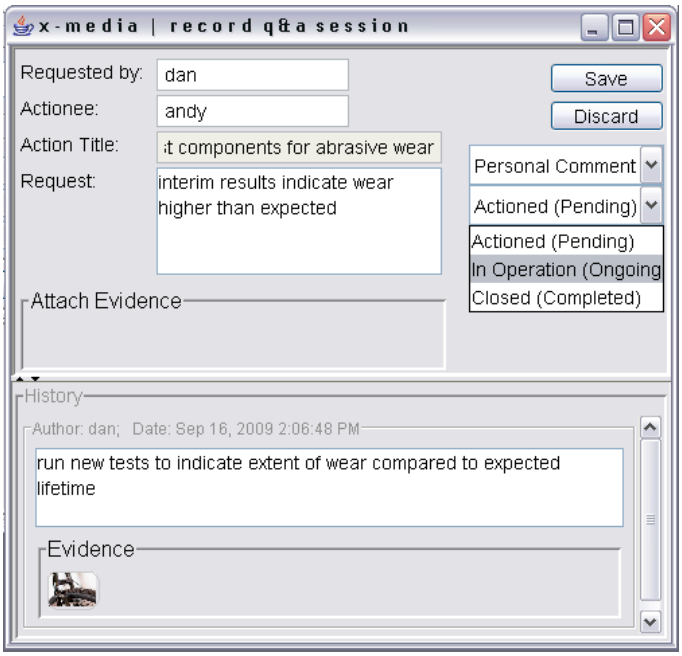


Figure 33. Record Q&A Session

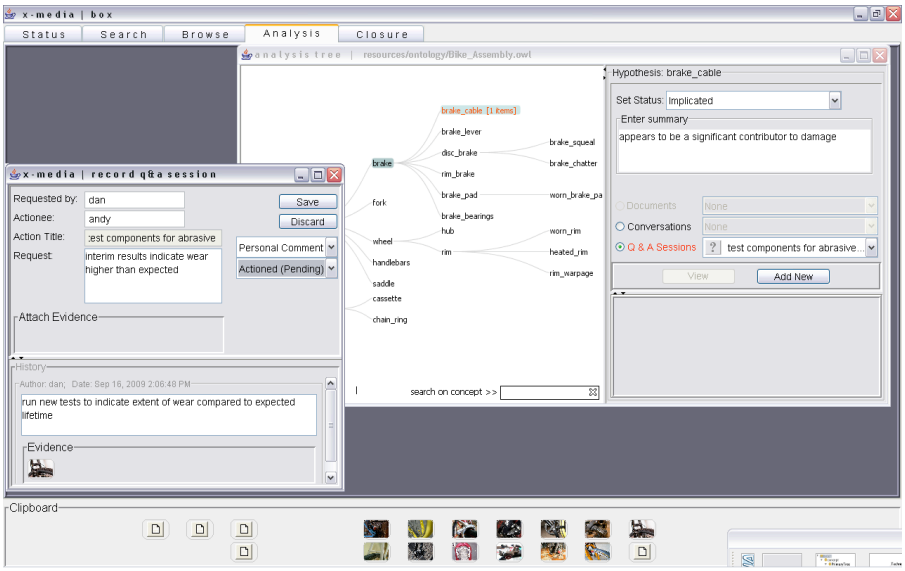


Figure 34.

Conclusions

In this deliverable we have described the current situation of the activities in work-package WP19 concerning the public use case. In particular, the X-Media technologies suitable for integration in the public use case have been briefly described, together with the adaptations needed to apply them to the public use case. The public use case is now not only defined in abstract but it can also be demonstrated practically using the XMediaBox GUI.

5 References

[Chen et al., 2006] J. Chen, D. Ji, C.-L. Tan, and Z. Niu. "Relation extraction using label propagation based semi-supervised learning". In *Proceedings of the Joint conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics (COLING/ACL-2006)*, Sydney, Australia, 2006.

[D1.8] Christoph Ringelstein, Sergej Sizov, "Formal Definition of Data Centric Process Knowledge", X-Media Deliverable D1.8, 2009.

[D1.9], Christoph Ringelstein, Sergej Sizov, "Prototype for Management of Data Centric Process Knowledge", X-Media Deliverable D1.9, 2009

[D4.3], Aba-Sah Dadzie, Thomas Franz, Yuangui Lei, Daniela Petrelli, Christine Preisach, "Evaluation of Knowledge Sharing Tools", X-Media Deliverable D4.3, 2008.

[D5.7] José Iria, Lei Xia, Alberto Lavelli, Lorenza Romano, Claudio Giuliano, Sebastian Blohm, "Final Library of Second Phase KA from Text Technologies", X-Media Deliverable 5.7, 2009.

[D6.5] Konstantinos Rapantzikos, Jenny Benois-Pineau, Christos Varytimidis, Spiros Nikolopoulos, Achille Braquelaire, Yannis Kalantidis, Yannis Avrithis, Jean-Philippe Domenger, Anastasia Moumtzidou, Yannis Kompatsiaris, Remi Vieux, Hugo Boujut, Elisavet Chatzilari, "Final Library for Image Analysis Technologies", X-Media Deliverable 6.5, 2009.

[D19.1] Lei Xia, Johannes Busse, Aba-Sah Dadzie, "Public Issue Resolution Demonstrator – Bicycle Use Case", X-Media Deliverable 19.1, 2009.

[Diday, 1971] E. Diday, *"Une nouvelle méthode en classification automatique et reconnaissance des formes, la méthode des nuées dynamiques"*, Revue de Statistique Appliquée, vol. 19, no. 2, pp 19-33, 1971.

[Ducheneaut & Bellotti, 2001] Ducheneaut, N. & Bellotti, V. "E-mail as habitat: an exploration of embedded personal information management", Interactions, ACM, 2001, 8, 30-38

[Franz et al., 2009] Franz, T, Scherp, A. & Staab, S., "Are Semantic Desktops Better?", *International Conference on Knowledge Capture, K-CAP,* ACM, 2009, 1-8

[Manjunath et al., 2001] B. S. Manjunath, J. R. Ohm, V. V. Vinod, and A. Yamada, “Colour and texture descriptors”, IEEE Trans. Circuits and Systems for Video Technology, Special Issue on MPEG-7, vol. 11, no. 6, pp. 703–715, Jun 2001.

[Zhu & Ghahramani, 2002] X. Zhu and Z. Ghahramani. “Learning from labeled and unlabeled data with label propagation”. Technical report, CMU-CALD-02-107, 2002.