PROJECT FINAL REPORT

Grant Agreement number: 258098

Project acronym: ALERT

Project title: Active support and reaL-time coordination based on Event pRocessing in open source software developmenT

Funding Scheme: STREP

Period covered: from 01/10/10 to 31/03/13

Name of the scientific representative of the project's co-ordinator, Title and Organisation:

Ljiljana Stojanovic, Dr., FZI

Tel: +49 721 9654 804

Fax: +49 721 9654 805

E-mail: Ljiljana.Stojanovic@fzi.de

Project website address: www.alert-project.eu

1. Publishable summary

1.1 Executive Summary

In one sentence, **ALERT is an open-source solution that helps developers stay focused on building better code**. In a bit more detail, ALERT is a research project within the EC FP7 framework that aims to improve coordination, collaboration and bug resolution processes taking advantage from building a project knowledge base in collaborative or distributed environments by providing methods and tools for **Software Development Companies**, **Open Source Communities** and **Developers** as well.

ALERT system revolves around 3 different key parts:

- **A distributed ALERT service system**, which is a cloud based system that holds the main architecture and functionality of all ALERT's components.
- **A set of "sensors"** that are required to proxy between ALERT system and the multiple different data sources and tools that ALERT system uses to collect information from.
- **User Application** that is the main interface where users interact with ALERT system.

The project has built a system that enables creating an Interactive Environment with developers that frees developers from accessing multiple tools for one single task. ALERT system collects information from multiple information data-sources; it then analyses collected information seeking for relevant events for developers; at the same time it also annotates & saves meta-data creating a Project Knowledge-Base accessible from the client based UI web application; and when it finds something relevant, it informs developers pushing them very customized and context aware notifications.

The main capabilities can be summarized as follows: **Automation & Notifications; Live Project & Team Status; Faster Team Interaction; Development Tools; Reduce Time solving Bugs; Multiple Data Sources; Talent Monitoring; Social Coding; Standard Ready** and it's **Open Source**.

ALERT has been built under an **Open Source Software (OSS) approach since its start**, and it is being developed with the participation, input and cooperation of well-known open-source communities (KDE and Linagora); large organizations (ATOS); European SMEs (CIM and Corvinno); and other relevant Academic and Research institutions (FZI, JSI, URJC, and ICCS).

ALERT, by producing a solution for the better open source development process, will have a strong impact on the increasing use of OSS in organizations (large companies and SMEs) and opens source communities; and the Research & Academic Community as well. It can also impact on other sector such as job recruiting services since it provides an automated way to create developers profiles to prove participation and experience; and show skills on Software based projects.

To best leverage its potential and future impact, **the project has selected an open-source solution that connects with its stakeholders through an open-source public model** that will profit from its adoption by one of the major open source communities in Europe and the world, KDE. It is of significance to highlight here the fact that a Spin-Off company created from URJC will exploit individually ALERT results to offer consulting services and built tools upon them.

1.2 ALERT's Challenges

Software development is an inherently collaborative, team based process, and hence requires coordination and control in order to make it successful, especially on the global economy where teams are geographically disperse and make massive use of disparate communication and collaboration tools. We have identified the following requirements for improving coordination in software development projects:

- **Integration of heterogeneous sources**: FLOSS communities use a variety of communication tools, including mailing lists for technical discussions and support, a bug tracking system for monitoring and resolving bugs and requests, or a source code repository for storing tracking changes in developed code. This makes the information pool very heterogeneous, distributed and not well-linked on the basis of its content. For example, the same problems reported in bug reports and discussion forums do appear to be isolated.
- **Interpretation:** Large teams and especially FLOSS communities generate a large volume of communication among its members and there are lots of discussion streams, unstructured messages and posted comments, which causes a kind of information overload. These posts can overload developers, interfering with and slowing their work. There is a lack of coherence that hinders developers in making sense of communications from others, or that produces unintended information filtering or misunderstandings. These interpretative difficulties in turn make it hard for team members to "capture" all aspects of the developing project, as it requires reading developer mailing lists, reading real-time chat, watching commits from the code repository, monitoring discussions about some bugs, etc., which introduce additional overhead.
- **Competence of developers**: Task assignment is still a bottleneck for open source software development. Many teams are largely self-organizing, without formally appointed leaders or indications of rank or role, raising the question of how the work of these teams is managed. In fact, deciding what features should be integrated in the release of the software, when and how to empower other code maintainers, or to find most "suitable" developer for fixing a bug, etc. is a challenge and inevitably introduces delays in development. Frequently, FLOSS developers can autonomously decide how and when to contribute to project development, following just some basic coordination. This makes the awareness about the work and needs from other developers very vague.
- **Notification**: One of the main issues in coordination is to enable that developers are proactively informed of other's actions. Developers/tools shall be "monitored" and notified as soon as something relevant happens. For example: if several end users/developers reported bugs with the same class, then the author of this class should know it as soon as possible (instead of "discovering" it at opening the issue tracking system).

Indeed, in order to participate effectively in the software development projects, developers need to understand who is working on what in the community and how their work affects other community members. This knowledge should be collected in real time, as it is created. Such knowledge would allow people to coordinate work effectively, anticipate other members' actions, discuss tasks and locate help. This is summarized in Figure 1.

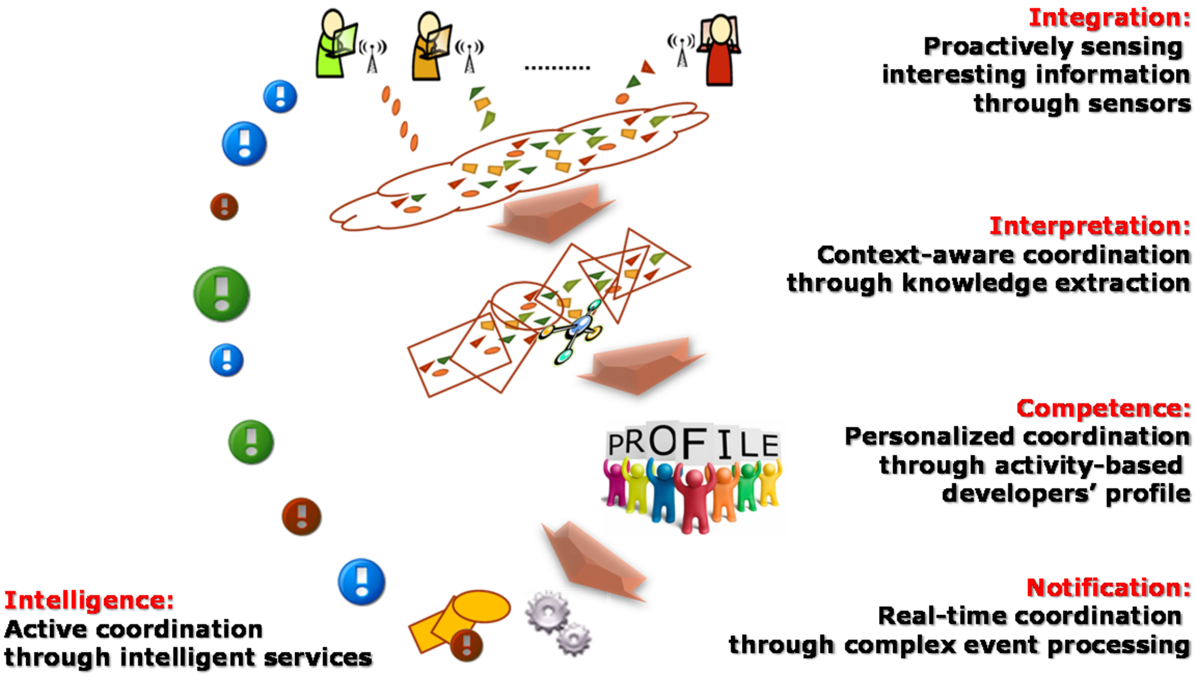

Figure 1: Summary of requirements for an active collaboration system

The main problem is that, in order to enhance coordination, communication must be efficient (i.e. timely and clear) and effective (i.e. relevant and targeted). In order to meet the above identified requirements, we proposed the ALERT system that maintains awareness of developers' activities through **real-time, personalized and context-aware notification**.

1.3 ALERT's Objectives

In order to address a solution for the previously cited real problems, the VISION of ALERT is to enable a new, more **interactive coordination** in software development teams, especially on the field of Open Source Virtual Teams that support **responding in a coordinated and more organized fashion to urgent, high stakes, and/or novel task demands being able to access relevant information from several data sources and use it to dynamically and interactively provide knowledge to developers**. This vision has been realized through the following set of objectives:

- **Efficient modeling of the more reactive** coordination in open source software development that will improve the awareness of the group work and the responsiveness of individuals, by (1) considering changes made in different (structured and unstructured) knowledge sources and (2) providing advanced services on top of them, including semantic integration and filtering of knowledge. This objective has been achieved through the delivery of:
  - ALERT **ontologies** that model the conceptual dependencies between community, content and interactions. Ontologies have been used as a backbone for defining the conceptual architecture of the collaboration platform. The ALERT Annotation ontology is dynamically created and extended based on the community content and is used for annotation of all available sources;
  - **Conceptual model of an (re)active collaboration platform** that not only facilitates communication and interaction within the open source community, but also functions as a virtual actor, processing the collected information/knowledge and interacting with the developers.

- **Efficient management of the awareness of team members** that enables interesting parties to be notified based on their interest/expertise as soon as something relevant happens without overloading them, interfering with and slowing down their work. This objective has been achieved through the delivery of:
  - Model, language and the PANTEON editor for **declarative specification of notification patterns** that enable representing the context during which (re)active behavior is relevant and the situations in which it is required;
  - **Intelligent, pattern-based engine ETALIS** that correlates, aggregates, and filters events in order to provide triggers for relevant notifications. We have taken a step further by providing a proactive reaction, that is, a reaction which is initiated before a certain complex event is recognized. We addressed this issue by providing a system which detects partially fulfilled events.
- **Efficient management of information** relevant for open source virtual teams, including the semantic integration of information and its flow between all stakeholders that supports better understanding of the situations which a developer should react on. This objective has been achieved through the delivery of:
  - **Module for integration of information from structured sources (KESI)**, extraction of metadata about community artefacts, community members and cross-linking of them. Metadata includes (a) references to code, files, packages, error traces, other bug reports and discussions, and (b) semantic concepts dynamically extracted and organized hierarchically in the Annotation ontology that represents a technical vocabulary or jargon that is meaningful to community members;
  - **Module for integration of information and extracting knowledge from unstructured sources (KEUI)**, semi-automatic annotation and bug duplicate detection in order to generate project digest. Concepts from external environment are linked thus making further processing of extracted information easier;
  - **Interaction highway set of services** that include novel, intelligent services offering support for automating critical tasks (like duplicated bug detection, task suggestions, and dynamic visualization of key descriptors of the project evolution). These services ensure more efficient and better management of issues submitted to the project.

- **Personalized and task-based access to information,** by allowing developers to focus on activities to be performed to achieve a specific shared task and/or by including information about the presence and activities of other developers in the open source community. This objective has been achieved through:
  - **Module for identifying developers across information sources and capturing the competence of the developer (STARDOM)** using (lightweight) ontologies for representation as well as groupings of developers based on different factors and the roles that they take in the community;
  - **Module for capturing relevant information for the task at hand (i.e. sensors for all available information sources).** This information is fed into the interaction highway in real-time and pushed to the developer.

The project consortium, coordinated by FZI, consists of 9 partners from across the European Union and it includes a balanced consortium of different types of leading organizations on the fields of academic and research, open source communities and both large companies and SMEs.

1.4 Highlights of Achievements

ALERT has obtained a considerable number of achievements in several areas and we encourage the reader to take a look with greater level of details on the documents and results that can be found on the deliverables produced within each of the work packages. We now cite some of the most important ones for conceptual and technical achievements.

1.4.1 Conceptual Achievements

The implementation of this project idea has provided the following non-quantifiable results in the form of best practices, guidelines, knowledge transfer and conceptual results:

- The ALERT **Conceptual Approach** is represented by the project idea and the conceptual architecture conceived for describing the proposed solution. The project idea rose from the perception in the OS communities that the management of the developers' coordination could be improved, and that a better process in bug resolution definitively would contribute to this improvement. This was later extended to consider not only OSS communities but also, all types of software development organizations, and solo developers as well.

  Thus, the innovation is not only to find the way to provide an answer to the communities' problem but also to find the right technologies to do it in a best way. The challenge to combine semantics, event-driven processing and information extraction techniques to implement the solution proposed becomes a result itself. It can be a proof of concept for the implementation of future solutions to similar problems in other contexts. The conceptual architecture described in D1.2 Conceptual Architecture deliverable shows a picture about how to build the solution and which components are required for that.

- The **ALERT Ontologies** model the conceptual dependencies between community, content and interactions. Ontologies are used as a backbone for defining the conceptual architecture of the collaboration platform. At building the ALERT ontologies, existing ontologies have been reused (such as QualiPSO, DOAP, FOAF, etc) and it is expected that further reuse of our ontologies will be done in the future by others modelling the same knowledge.

- The **ALERT Pilots** are three based on the three OS communities represented in the project: KDE, OW2 and Morfeo. These three communities have identified room for improvement in the bug resolution process and have showed that ALERT software can offer a way to increase efficiency in development process and management of development teams.

  The conclusions and recommendations produced after these trials propose interesting results and best practices that can be later reused by other teams and companies seeking to improve development processes. This information will be described in D8.3 Evaluation Report due at M30.

- The **ALERT Knowledge Transfer** by producing several publications during the project life by research partners. Besides its relevance to dissemination of RTD activities of the project they can also serve as reference for further research work.

1.4.2 Scientific and Technical Results and Foreground

Software components are the most relevant and complex result of the project. During the project, ALERT consortium has provided two major software releases with incremental functional enhancement. In order to implement the desired functionality, ALERT is structured as a set of components:

- Sensors that are used to monitor all information sources relevant for the bug resolution and notify about a change;
- Knowledge extractor from structured sources (KESI) that is used to extract information from the structured sources;
- Knowledge extractor from unstructured sources (KEUI) that is used to extract information from all unstructured data;
- Metadata service that is used to create, store and trigger semantically-enriched information and provide query and reasoning facilities. Metadata service also provides support for creation, retrieval, reasoning about and storage of ontology-based metadata;
- Profiler (STARDOM) that is used to identify developers and create/update their profiles based on their activities;
- Interaction pattern editor that is used to enable developers to describe situations relevant for notification;
- CEP engine that is used to discover interesting situations;
- Action service that is used to execute actions related to the detected situations;
- Search service that is used to enable users to specify their information needs and provide support for detecting different relationships between the bug reports;
- Recommendation service that is used to suggest a most competent developer to resolve an issue and a set of issues that are most suited to the developer’s competence;
- Annotext and Ocelot components that are used for ontology creation;
- Visualization service that is used to provide different visual perspectives on the provided results;
- The ALERT web UI that is used to interact with the end-users.

The picture below shows all the components and the interactions among them:

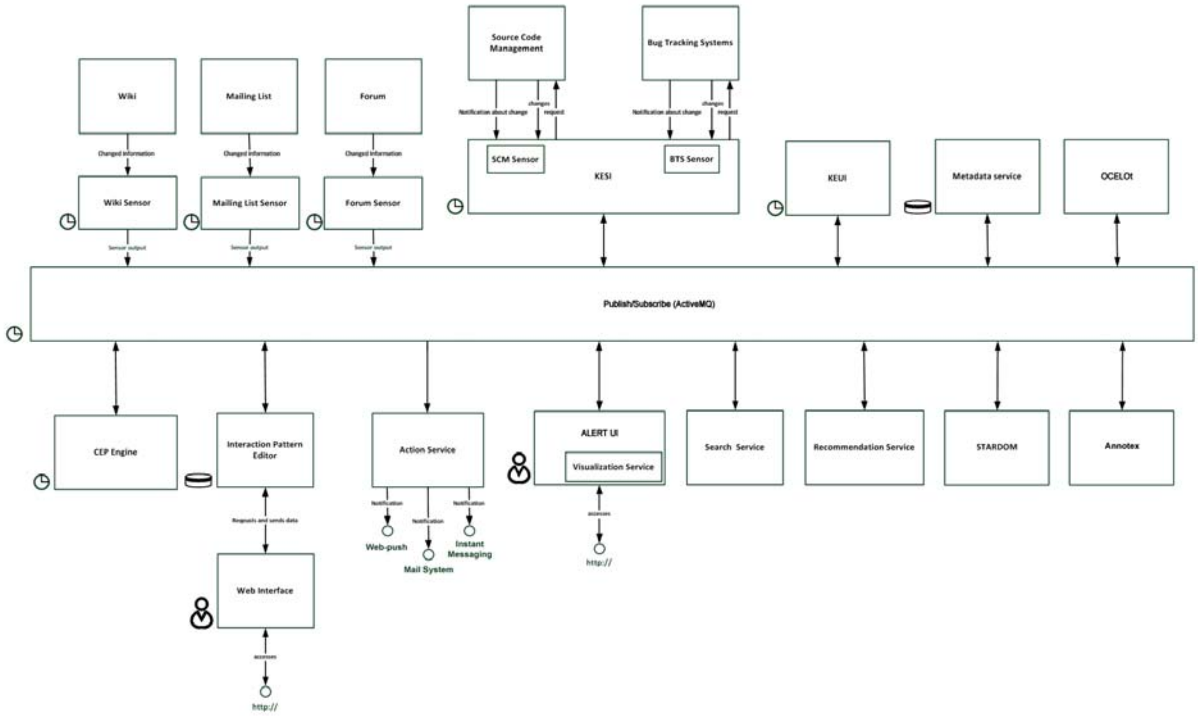

Figure 2: ALERT Architecture

The complete description of components and the conceptual architecture description can be found in D1.2 Conceptual Architecture deliverable, D7.2 Integrated ALERT system & System Manual and D7.3 Evaluated ALERT system.

In the rest of this section we discuss the main achievements.

1.4.2.1 Progressing state-of the art in integrating/extracting knowledge from structured sources

Some of the most important data sources with information about the software development process in modern software forges include many structured information. Notably, both source code management systems and issue tracking systems (ticketing systems) store detailed and structured information: in the case of source code management, in the form of commit metadata; in the case of issues tracking, in the form of tickets and changes to tickets. Both of these kinds of repositories are important for ALERT, and are indeed the ultimate source for a large amount of the information managed by the system.

Retrieval of data from this kind of repositories is not new. The academic community is mining it since at least the early 2000s, and during the last years of the decade, some tools specifically devoted to that task do appear. Among them, the Alert project selected the LibreSoft Tools toolset (currently renamed as MetricsGrimoire), maintained by the team from URJC. When ALERT started, they were probably the most mature free / open source toolset in this realm. During the duration of the project it has continued evolving, nurturing a community around the tools, and getting commercial support from Bitergia, a start-up in the business of software metrics.

The tools from MetricsGrimoire that have been used and improved in ALERT are:

- CVSAnalY, which extracts information from source code management systems: CVS, Subversion, git, and Mercurial (through a git bridge).

- Bicho, which extracts information from issue tracking systems: Bugzilla, RedMine, Jira, and the ticketing systems provided by SourceForge (including the newly deployed Allura software), GitHub, Google Code, Lanuchpad, and other forges.

When the project started, these tools already provided support for batch data extraction of the past history of the repositories. But this functionality fell short in fulfilling the requirements imposed by ALERT, as did other tools available at the moment. The most strict requirement was the need of tracking all new events, both in the source code management system and in the issue tracking system, as they happened (or with a very short delay). This is needed so that the corresponding information could be incorporated as soon as possible into the ALERT knowledge base, with all ALERT services being notified, and being able to update their results with the new data.

For satisfying this requirement, and for injecting the resulting information into the ALERT bus (needed so that it could reach ALERT services), two main actions had to be performed:

- A rearchitecting of CVSAnalY and Bicho, so that they could work incrementally and continuously, reacting to new commits, new tickets and changes in tickets as they happened.
- A new component, KESI (Knowledge Extractor from Structured Sources), that is an integral part of the ALERT architecture: the core component for the extraction of knowledge from structured sources in ALERT.

CVSAnalY had to be redesigned so that it could incrementally retrieve new commits, updating all its internal data structures, and feeding the new information to KESI, that would produce the events expected by ALERT services. This way, CVSAnalY has been upgraded to be a mature and proven tool for both batch and real-time processing of the commit metadata and commit-related metrics (such as size or complexity metrics of new revisions of files). The capture of the new commits was done via post-commit hooks, available in most source code management systems.

The case of Bicho was more complex, since it not only had to be redesigned to capture new tickets and changes to tickets as they happen, but also to incrementally reconstruct the whole change history of tickets, for batch-production of events, used when initializing the ALERT system. This again led to a more mature tool, and to be unique (to our knowledge) in extracting both the complete history of an issue tracking system, and that related to new events happening in it, as they happen. In addition, the backends for Jira, GitHub, and FusionForge had to be completed, since those were fundamental for some of the ALERT case examples.

Both tools were also stressed to their limits, showing how they were capable of handling very large repositories, such as those of KDE or Eclipse, in the context of the project, or WebKit, GNOME, or OpenStack that were also run for testing stability and maturity. This means that MetricsGrimoire have advanced the state of the art in retrieval information from software repositories by adding new ways of extracting that information, by supporting a comprehensive set of repositories, and by adding the capability of retrieving their information in real time.

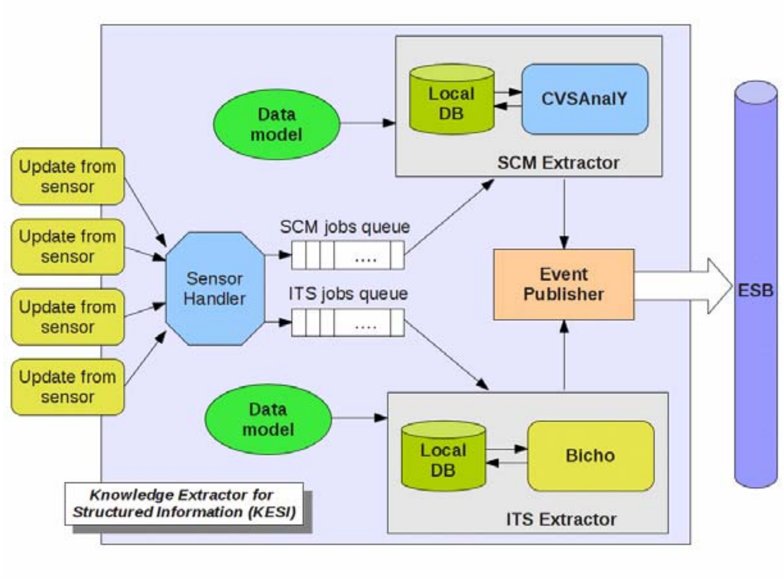

Figure 3: Internal structure of KESI

KESI was developed specifically for the ALERT system to fulfill the following tasks:

- Getting information from structured repositories, using CVSAnalY and Bicho as the lower level tools. These tools feed the KESI database, which is the origin of all structured information in ALERT.
- Coordinating the activity of sensors deployed in the data sources (repositories), which forward to KESI raw data about new events as they become available, instructing CVSAnalY and Bicho to parse it.
- Interacting with the rest of the ALERT systems through the common event bus.

The idea behind the design of KESI was to reuse as much as possible the existing MetricsGrimoire tools, so that the effort was achievable with the resources available in the project, and the focus could be put on the new requirements imposed by it. This fact is reflected by its internal structure:

- Sensors are in charge of sensing changes (new commits or changes to tickets) in the tracked repositories.
- The information produced by those sensors is routed to the MetricsGrimoire tools (CVSAnalY and Bicho), which interpret it and produce the needed changes to their internal databases (which all together form the KESI database).
- Once they are done, the Event Publishers extracts the relevant information from this database, and produced ALERT events, that are injected in the ALERT event bus, so that they can reach services interested in them.

In addition, KESI supports the two main modes supported by ALERT services: offline mode, when it retrieves historical information from the tracked repositories, requesting a complete complete log of all the activity that took place in the system; and online mode, receiving live updates about any changes performed on the tracked repositories. KESI (in combination with CVSAnalY and Bicho) has shown how it is capable of handling repositories containing histories with hundreds of thousands of relevant events in offline mode, and processing several tens of events per second in online mode (assuming no delay in the event bus).

In summary, although the basic technology for retrieving information from structured sources was available at the beginning of the project, it has been significantly upgraded and completed during its lifetime, both in terms of maturity, support for diversity of platforms and reliability, and in support to the specific needs of a real-time event system such as ALERT. This has allowed the project to produce the most complete and comprehensive technology for these tasks available today.

1.4.2.2 Progressing state-of the art in integrating/extracting knowledge from unstructured sources

Content that is generated in different information channels can be considered as structured or unstructured. In case of source code commits, for example, the commit information is mainly structured since it clearly identifies the files that were modified as well as the methods that were added, removed or modified. Similarly, an issue created on the issue tracking system contains a lot of structured metadata such as the author of the issue, the product and component the issue is assigned to, the status and resolution of the issue, etc. The majority of content generated in information channels monitored by ALERT is however plain text created by users. Text itself has no direct value for the computers which is why we have to perform knowledge extraction in order to obtain from it relevant semantic information.

In ALERT we perform three different types of knowledge extraction from text. First type of extraction is cross-source linkage which identifies in text references to related posts, possibly in other information sources. Secondly we perform semantic annotation of terminology relevant for computer science and software development. The third type of knowledge extraction is representation of text in a metric space that can serve for tasks such as finding related posts or possible bug duplicates. Each of these types will now be described in details.

Cross-source linkage

Posts frequently contain references to other related information. An example could be a remark in a forum post that the described problem could be related to a known issue #123 in the issue tracking system. By identifying such references we are able to provide the user of the ALERT system an extended view on a particular topic that is not limited to a specific information source or even a specific discussion (thread). Instead when checking a discussion or an issue, all related discussions can be immediately brought into view thus allowing the user to expand his knowledge of the topic. In order to identify in text references to related resources we use a set of regular expressions. Using these expressions we are able to detect all URLs as well as patterns such as “bug #123” or “issue #123”. Such references are then annotated, disambiguated (the unique identifier for the referenced resource is found) and stored in the knowledge base.

Frequent references in text are also references to source code. Since we are also monitoring source code management systems we are aware of all files, classes and methods developed in the project. Whenever there is, for example, a stack trace provided in an issue report, we can automatically annotate in it the references to files, classes and methods. These references are again stored in the knowledge base and are used in use cases such as finding appropriate developers to fix an issue and finding issues possibly caused by a developer’s code commits.

Semantic annotation of text

A crucial part of knowledge extraction is also understanding of what the provided text is about. In order to extract semantic meaning from text we created an Annotation ontology that is used for annotating all text processed by the ALERT system. Since ALERT is to be used by software development communities the developed Annotation ontology contains all domain knowledge related to software development and computer science in general. Each concept from the ontology can have one or more labels and whenever the label is detected in text, we annotate the text with the concept. Using these annotations we are able to perform tasks such as finding posts with similar content and performing semantic search. Both of these tasks rely on the fact that different terms that represent the same thing are in the ontology represented with the same concept and that different relations can exist between concepts. As an example, consider that the user performs a keyword search for “dialog”. The concept of the dialog can be represented with several labels, such as “dialog”, “window” or “form”. Instead of returning results that only directly mention the word “dialog” we can also include posts that mention other two labels. Additionally, since we are searching using the actual concepts we can also exploit the relations between the concepts. When searching for one concept we can therefore also consider returning results containing some closely related concept. In KDE domain, for example, searching for “email client” could also return results containing “KMail” which is KDE’s email application.

The key element in semantic annotation is the Annotation ontology and we will describe now in more details the process of its construction. The process consists of two main steps – (a) identifying the concepts related to computer science, and (b) identifying the relationships between the concepts. Since each software project also introduces some of its own terminology (product names, dialog names, etc.) we will also describe two developed approaches for extending the Annotation otology with project specific terminology.

Identifying concepts for Annotation ontology

To find relevant terminology we identified online glossaries related to computer science. Two most up-to-date websites that we found were *webopedia.com[]* and *whatis.techtarget.com[]*. From both sites we extracted all the terms as well as term descriptions which we later used for extracting relations between concepts. For identifying terminology that is very specific to software development we used the *stackoverflow.com* [] website which is a Q&A system with more than 2.5 million questions related to software development. The site has an extensive set of tags used for tagging the questions, and we used the most popular 2000 tags as terms for the ontology.

After obtaining the starting set of terms our first goal was to merge the synonym terms that should be represented with a single concept. One way of merging terms was done using *stackoverflow*’s list of synonym terms which contains around 1400 synonym pairs. Second way of merging was done using the term descriptions inside which we searched for patterns such as “X also known as Y” or “X (abbreviated Y)”. When such patterns were identified, X and Y can be considered as synonyms and can be represented with the same concept. In this way we were able to obtain a set of concepts where each concept has one or more labels that represent it.

As a next step we wanted to link the concepts to the corresponding Wikipedia articles. Using this connection we would be able to extend the set of concepts as well as obtain additional information about the existing concepts. To find the appropriate Wikipedia page for a concept we matched the labels of the concept to titles of Wikipedia articles. We identified two challenges with this approach. The first challenge is that the Wikipedia page for a label can be a disambiguation page. This occurs when a label has several popular meanings. An example of such label is “TTL” acronym which can have several meanings such as *Time to Live*, *Transistor Transistor Logic*, *Taiwan Tobacco and Liquor*, etc. To avoid manual disambiguation we simply skipped connecting concepts with such labels to Wikipedia pages. The second challenge were labels that are so frequently used in common language (outside computer science) that they are not even considered ambiguous. An example of such label is “ant” which in computer science refers to Apache ant (a tool used in automatic build process) but is mapped to Wikipedia article on the ant insect. To remove mistakes of this type we manually checked the list of matched Wikipedia articles and removed the ones that were invalid.

After mapping labels to the corresponding Wikipedia pages we used the content of these pages to identify new terms which were not covered by the glossaries. To do this, we only used the first paragraph of each article, which typically gives a short definition of the term. These paragraphs also frequently contain links to articles describing closely related concepts. We used the articles linked in the first paragraph as candidates for new terms and sorted them by their frequency. Our assumption is that if an article is linked to from many articles related to computer science, then the article is also likely about computer science. Based on this assumption, titles of frequently appearing articles were added to the ontology as new concepts.

Using the described approach we were able to automatically obtain 10,233 labels related to computer science and they are organized into 6,196 distinct concepts.

Extracting relations between concepts

Relations between concepts are an important part of the ontology. As we mentioned, relations can be used for use cases such as semantic search where we wish to expand the search to include related concepts.

In order to create relations between the concepts we can use the term descriptions that we obtained from online glossaries and Wikipedia pages. These descriptions usually contain several hyperlinks to other related terms and these links can be used to automatically create relations between the corresponding ontology concepts. The limitation of using hyperlinks is that they contain no semantic information about the type of relation so we are only able to create some general type of relation. In our ontology we represented them using *linksTo* relation.

In order to obtain more specific and usable concept relations we decided to apply natural language processing (NLP) methods on term descriptions. We first used Stanford parser [20] to obtain a dependency parse of the sentences. The dependencies provide a representation of grammatical relations between words in the sentence. For the next steps we only consider sentences in which two known ontology concepts co-occur. Using Stanford Part-of-Speech (POS) tagger [21] we then tagged the words in each of the remaining sentences and identify the verb that is used to describe relation between the two mentioned ontology concepts. As a result we obtained triples such as:

“WSDL”, describes, “Web service”, “Microsoft”, created, “Windows”, “Apple Inc.”, designed, “Macintosh”

Using Wordnet [] we were then able to group obtained relations by synsets (sets of synonyms). From all the sentences we obtained verbs that can belong to around 750 different WordNet synsets. Of all these synsets we only chose those that we were able to map to relations *isPartOf, hasPart*, *creator* and *typeOf*. We chose these relations because they are mostly hierarchical and are most useful for expanding the search conditions. The final number of relations that we were able to extract from the text was 91,122.

Extending the Annotation ontology with project specific terminology

Each software development community that will be using the ALERT system also has its own vocabulary containing terms related to the developed project. In order to identify and add this terminology to the Annotation ontology we developed two approaches that will now be shortly described.

The first approach relies on statistical analysis of existing text. In this case, the available text from information sources is transformed and represented using the vector space model. Using different clustering methods we can then identify clusters of related posts. From each cluster we can identify the most relevant keywords and provide them to the ALERT administrator for further inspection. The terms that are validated are then added to the Annotation ontology as new concepts. The tool that can identify candidates for new concepts is called Annotex and can along with the candidate terms also provide suggested description of the terms and a list of existing concepts that are possibly related to each candidate term. The advantage of using the statistical approach is that it does not make any assumptions regarding the text language and can therefore be applied on text in any language.

The second tool we developed is called OCELOt and uses linguistic techniques to identify project specific terminology. The tool uses Stanford Part-of-Speech tagger to detect noun phrases in text.

Noun phrases that occur frequently enough are then shown to ALERT administrator who can decide to add them or not to the ontology. The advantage of this approach is that it performs text analysis and therefore suggests noun phrases (and not general terms) that are most likely to be added to the ontology. The downside is, however, that it only works on text for English language.

Information extraction for bug duplicate detection

Beside semantic extraction from text we also need to transform the text into a representation that allows us to perform tasks such as finding posts with similar content. Finding similar content can have different uses but in case of software development we are most interested in using it for detecting bug duplicates.

Bug duplicates are bug reports that are describing the same issues as some previously created bug report. Such reports are an issue because, if undetected, they result in unnecessary work since multiple developers are assigned to work on fixing what is essentially the same issue. Before assigning an issue to a developer, the bug triager therefore has to manually determine if the issue is a duplicate or not. This requires from bug triager an extensive knowledge of all reported issues. In large communities, such as KDE, that currently has more than 300,000 reported issues, this can be a daunting job. Even more discouraging is the fact that duplicated bug reports are very common. In KDE, for example, every fifth report is marked as a duplicate. To make the task of bug duplicate detection easier we therefore want to provide tools that for a given bug report quickly and accurately identify most similar reports. Using the ranked list of similar reports the bug triager should be able to more efficiently determine if the bug is a duplicate or not.

As the basis for determining if a bug is a duplicate or not we decided to use text similarity. In order to be able to compute similarity between reports we have to perform a series of preprocessing steps. First we need to decide what text to use when computing the similarity. A bug report can contain a subject, a long description of the issue and possibly a number of comments. In our experiments we evaluated performance by using or ignoring different parts of the reports and found that the best accuracy can be obtained using all available text – subject, description and all comments. For detecting bug duplicates we therefore extract all these parts and merge them into a new document that will now represent the bug report.

Next we need to remove from the document the stop words and stem the words using Porter stemmer [22]. Stop words are very common words that don't convey any meaning, such as "and", "an" and "by". Stemming is a process of putting words into their base form by removing word inflection.

In the last step, the document is transformed into vector space model. The vector space model [23] is a widely used technique in information retrieval. In vector space model, each document is represented as a V-dimensional vector, where V is the number of unique terms (size of the vocabulary) that appear in all the documents. For a document d_j, the corresponding vector $\overrightarrow{d_j}$ in vector space model looks like

$$\overrightarrow{d_j} = [w_{1,j}, w_{2,j}, \dots w_{V,j}].$$

The weights $w_{i,j}$ correspond to TF-IDF weights that are computed as a product of term frequency (TF) and inverse document frequency (IDF). Using such weighting scheme, the weight assigned to a term t in a document d is highest when t occurs many times within a small number of documents (the term is very discriminating for those documents). The weight is smaller if the term appears in many documents or occurs fewer times inside a document.

After the documents are transformed into vectors, similarity between two documents can be computed using a vector similarity measure. A commonly used measure that we also used in our experiments is the cosine similarity.

Using the bug reports represented in vector space model and the cosine similarity we can now create for any bug report a ranked list of most similar bug reports. For each report in the list we also get a numerical score of similarity in the range between 0 and 1. Using this list of similar reports, the bug triager can check a small subset of most similar reports in order to decide if a bug is a duplicate or not. In order to evaluate the accuracy of bug duplicate detection we tested the methods on bugs from KDE bug tracking system. The repository was provided by KDE community in November 2010 and contained 249,129 bug reports of which 47,061 were marked as duplicates.

In order to evaluate the accuracy of the method we needed to select a scoring function. The function that we decided to use is called *recall rate* and is computed for a certain size of the ranked list of issues. It was suggested by Runeson et al.[Runeson] and is defined as

$$recall\ rate_n = \frac{N_{recalled,n}}{N_{total}}.$$

In this formula, the N_{total} is the total number of duplicated reports and the $N_{recalled,n}$ is the number of duplicates that we were able to detect by looking at n most similar issues. In other words, the measure tells us what proportion of duplicates can be successfully identified by looking only at the suggested list of reports of size n. In Table 1 we present recall rates for n values of 1, 5 and 10. By looking at the »Content similarity« column we can see that about one third of all duplicates can already be identified by checking only the most similar bug report. By checking 5 most similar bug reports we can identify almost half of all duplicates. These results are rather impressive if we consider the fact that we are ranking almost 250,000 bug reports.

	Content similarity	**Content similarity + time difference**
recall rate (n=1)	32,42	35,76
recall rate (n=5)	48,73	53,46
recall rate (n=10)	55,18	60,53

Table 1: Recall rate obtained using content similarity or content similarity combined with time difference between bug reports

Despite good results we were interested to see if the accuracy can be improved by cleaning up the content of bug reports or by using some other features beside the content. We performed several experiments where we tried (a) removing spam reports, (b) removing stack traces, (c) removing common words, (d) using different parts of bug reports (subjects, descriptions, comments), (e) filtering of reports based on product/component information, (f) reranking reports based on time difference and (g) reranking reports based on hubness. Detailed discussion and evaluation of these experiments was described in deliverable D3.4., but we will shortly describe the experiment with reranking based on time difference between the compared bug reports. By inspecting the time differences between the original bug report and it's following bug duplicates we noted that bug duplicates are very often created soon after the original bug report. More specifically, 1/3 of bug duplicates are created less than 5 days after the original bug report. Based on this information we decided to update the ranked list of most similar issues by taking into account the time difference to the tested issue. Issues with small time difference were moved more to the front of the list while issues with large time difference were moved more to the back.

By applying such reranking we were able to achieve about 6% improvement in recall rate (see second column in Table 1.).

1.4.2.3 Progressing state-of the art in developer identification and developer profile modeling

The first step of the process profiling, is that of identification. Gauch et al. [1] state that the initial provision of data for identification may be collected explicitly or implicitly. Explicitly means when the user has direct intervention providing information such as a registration form. Implicitly is through methods such as agents whilst using the system.

Carmagnola et al [2] defines identification as a means of authenticating the user when accessing the system, this can be implemented trivially via a login form, or through more implicit methods such as the use of cookies or session ids.

Explicit methods for identifying a user such as a login form do not necessarily need to be in single system environment. Identification of a user can be done across systems. In cross-system identification different properties relating to the same user can come from different sources. The same user can provide different information across systems, meaning that each property should be weighted accordingly when used for comparison to identify a single user. For example in the Mozilla bug tracking system only the e-mail is used to uniquely identify a developer[3].

Authors in [2] analyze the importance of each profile property, for example the last name has a higher probability of being accurately provided by the user than the birth city and propose the conceptualization and implementation of a framework that provides a common base for user identification for cross-system personalisation among web-based user-adaptive systems. It represents a hybrid approach which draws parallels both from centralized and decentralized solutions for user modelling. To perform user identification, set of identification properties are compared using an identification algorithm.

This can be extended with the use of existing information across other existing open identification systems. F. Abel et al. [4] experiment with augmenting a user's profile for identification by using the information found in social profiles and existing social ID's, for example through the use of OpenId.

A more implicit approach of identifying a developer is that provided by Lange et al [5] which examines author identification via source code style, using code metric histograms and genetic algorithms in order to identify code authorship. Code style is determined as any decision over which the developer has discretion.

Iváncsy et al [6] present a system called WAT (Web Activity Tracking system) which mines log files to identify a user by tracking his behaviour using a combination of information gathered through complex cookie data collection and log mined information.

Robles et al. [7] propose an approach, based on the application of heuristics, to identify the many identities of developers where identification information is gathered from different sources, and a data structure for allowing both the anonymized distribution of information used for identification, and the tracking of identities for verification purposes.

STARDOM

In ALERT we provide **STARDOM** a **SofTwARe Developer cOMpentency profiler**, which bases the profile of the developer on a competency model. This model comprises four weighted *attributes* that are calculated from metrics extracted from the activity of the developers in the tools used by the community. The four attributes are Fluency, Contribution, Effectiveness and Recency.

Fluency states the degree to which a developer has general knowledge about the different aspects of the project, such as the members of the community, the different modules that make up the system, the internal tools and languages used for development, etc [8]. A metric that is used for the calculation of the fluency of the developer is *the number of API usage counts* [9]. **Contribution** denotes the volume of work the developer does for a specific project, in terms of source code contributions, patches committed, activity in mailing lists, forums and issue tracking systems [10]. Contribution takes into account metrics such as *the communication activity* (which estimates the volume of information exchanged via mailing lists, forums or both), *the ITS activity* and *the SCM activity*. **Effectiveness** is a measurement of the effect of the developers contributions, calculated from metrics such as the *number of issues fixed in the ITS*. Last, **Recency** is a measurement based on the *time* when the last activity has been recorded for a developer in any of the information sources, such as *the time of the last communication contribution*, *the time of the last ITS action* and the *time since the last commit*.

In order to be able to calculate the developer competency we need to derive both the weights of the metrics in relation to the attribute they are under as well as the weights of the attributes in relation to the final competency index. In order to do so we examined different methods, and concluded in using a Naïve Bayes classification process for expert developers.

The first step of the process is to extract metrics (SCM activity (SAM), ITS activiy (IAM), Communication Activity (CAM) and number of API Introduced (NAIM)) from the entire history of the project from three main information sources, the **ITS**, the **SCM** and the **Mailing List**. This resulted in metrics derived from 104K commits, 1K issues and 900 mailing list messages since 1999 to date. This extraction process resulted in more than 2000 different developer profiles throughout the projects history.

In the second step, for each metric extracted we created a ranked list of developers, where the developers with highest values are ranked first. We then composed a list of 80 developers by joining the top 20 developers in each list and requested feedback from experts in the KDE Solid community. The result of this process was a list of the *current* top 11 developers in the community.

Finally, using this feedback, we created a set of users with 2 classes, the “Core Developers” which included the top 11 developers, and the “Community Members” which contained the rest of the community developers (2116 developers). Based on this set, a Naïve Bayes classifier was built, assuming normal distributions for the numeric values of the metrics. For the “Core Developers”, the class description can be found in Figure 4.

These normal distributions can be used to model the impact of each metric to the final calculation of the CI. These distributions are also used to distinguish competent core developers based only in their metrics.

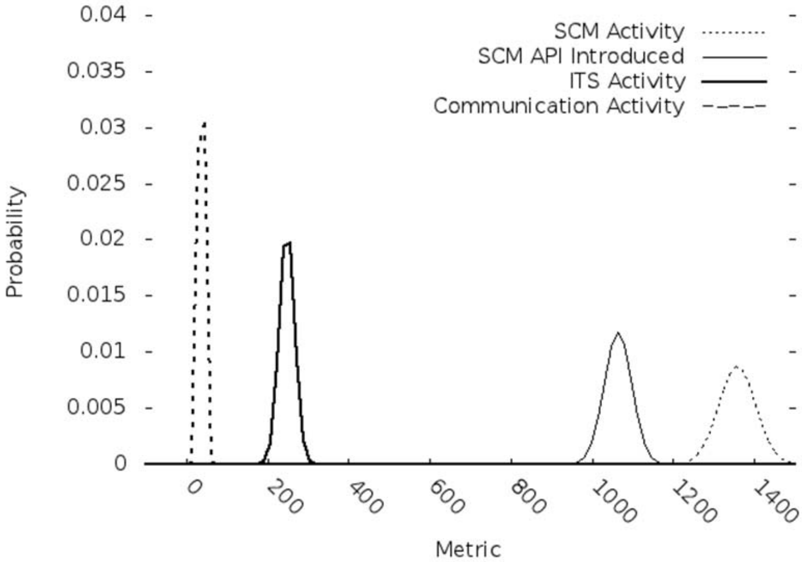

Figure 4: Class Description for "Core Developers"

Groups of Community Members

In addition to Core Developers, which are defined as the most competent members of the community in bug resolution, we were able to identify other classes of community members. These classes are the following:

- **Prospects:** This group consists of people in the community that show promise based on their activities as future core developers. Therefore it is valuable for the community to engage them in the development process.
- **Bug Triagers:** This class refers to developers that apply a bug triage process, in which each new bug is assigned a priority based on its severity, frequency, risk, and other predetermined factors.
- **Testers:** This class contains contributors that participate in validating and verifying that software works as expected and complies with a number of same characteristics.
- **Occasional Contributors:** Members of this class are occasionally contributing source code or other resources to the project.
- **Reviewers:** People in this class are re-examining the components that are built by somebody else. A reviewer is someone who is knowledgeable on the interfaces and knows how the product works.

1.4.2.4 Progressing state-of the art in event processing

In order to enable real-time detection of situations in ALERT, we built an infrastructure for complex event processing (CEP) upon the event-driven architecture of the whole ALERT system. Event processing techniques are mainly used in domains where certain patterns of events trigger some action in near real-time. Often these systems are built around events where applications analyze and react to events. The conceptual model for complex event processing in ALERT is illustrated in Figure 5. The main components are:

- An interaction pattern editor which allows the definition of situations via a graphical user interface;
- A CEP engine that provides the complex event processing and stream reasoning functionality;

- An action manager aiming at the subscription to defined situations via different communication channels;
- A pattern transformation module which translates the RDF-based pattern definition language in the editor to logic-based CEP rules.

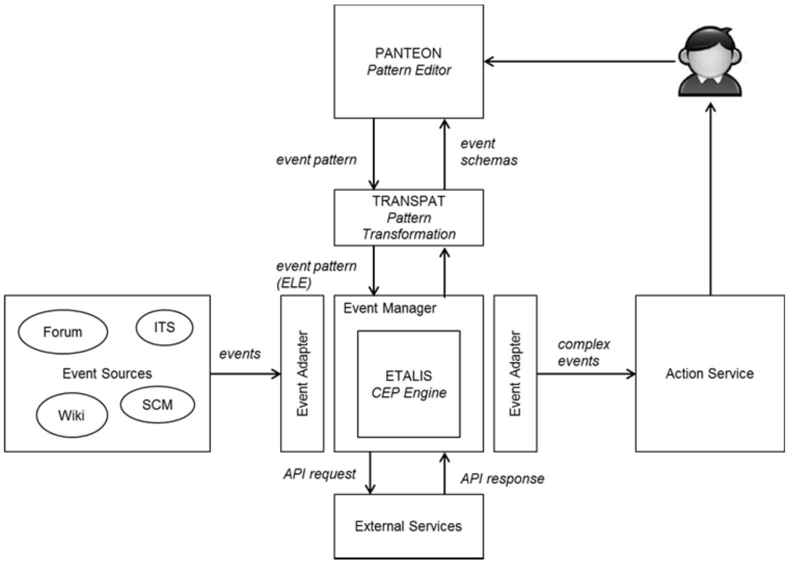

Figure 5: Conceptual architecture of CEP in ALERT

Innovation related to the design-time aspects

Complex Event Processing (CEP) is the analysis of events from different event sources in near real-time to generate immediate insights and response to changing business conditions [25]. The main strength of CEP is to derive high level events from a set of input events. Event pattern matching is the most powerful capability of CEP [26]. Event patterns encode the knowledge for a relevant business situation that is of interest. A relevant business situation can be the detection of bottlenecks, process failure or customer dissatisfaction. Luckham [25] considers the event patterns as the foundation of CEP systems. From the business perspective event patterns represent the most valuable asset in CEP. Event patterns are used in order to derive high level complex events describing a relevant business situation.

While nowadays CEP systems provide an efficient and effective infrastructure to detect event patterns their management is still in its infancy. Compared to other IT-systems, CEP systems still lack in support of tools and methods allowing users to reconfigure a system easily or to refactor services and components [27]. The main focus of the nowadays CEP research is focused on the efficient matching of a high volume of events in a short time period and disregard the role of supporting software tools for event pattern management. Although most of the commercial solutions provide a graphical environment to deal with event patterns they are not based on fundamental methodologies, methods and techniques to cover the life cycle of an event pattern.

Since the need for near real-time situation detection is increasing there is a potential need for tools and methods dedicated to efficient and effective event pattern management.

The current generation of event processing tools is rather programming oriented where fundamental programming skills are needed to develop event patterns [26]. However, there is a demand towards easy to use tools allowing users, who might not have deep programming skills, to define, deploy and maintain event patterns.

Luckham describes in his article [30] the problem with event pattern rules in CEP systems. Especially when the numbers of event processing rules increase their management is a challenge.

The problem arises since the languages in which an event pattern is written are incomprehensible. The result is that it is hard to read and understand the event pattern rule set. Therefore he proposes a high-level rule language that can be used to express rules succinctly in a way that makes them understandable. He describes the event pattern rule language as the first step in event pattern rule management. The focus is more on human readable event pattern rules. He defines event pattern rule management as writing correct event pattern rules, organizing event pattern rule sets for efficient execution, making changes correctly and ensuring logical consistency and absence of redundancies.
Turchin et. al [31] describe the problem of event pattern definition by domain experts. They describe the issue that it is a hard task to let the domain experts to define all elements of an event pattern. Therefore they propose a framework for automating the task of specifying event patterns by combining the knowledge possessed by domain experts with automatic techniques for specification of rule parameters.
The EPTS (Event Processing Technical Society - http://www.ep-ts.com) which is a diverse community including organizations from research and industry recognized the importance of event management and defines it as a discipline that encompasses among other things, the design, the development and the maintenance of events. It is even more complicated in the case of event patterns, as an event pattern consists of several events related by different operators.
The main innovation features are:

- **Management life cycle:** The first step towards an efficient and effective pattern management is having the understanding of the event pattern life cycle. Similar to the software development process we defined the life cycle of an event pattern as a process covering the modelling, refinement, execution and evolution of event patterns. The life cycle consists of three main phases that cover both design-time issues and run-time issues. These phases form a feedback loop enabling continual flow of information collected in the entire life cycle of an event pattern. We refined each phase with a set of tools and methods in order to provide the pattern engineer additional support.

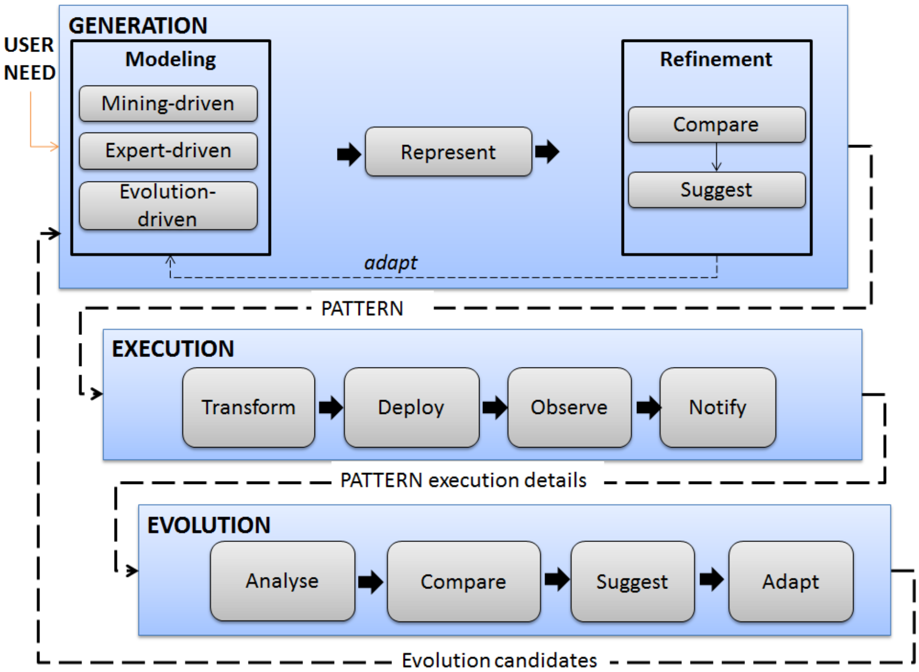

Figure 6: Interaction pattern life cycle phases

- **Execution monitoring:** Current CEP engines are designed as a black-box that matches a pattern against incoming events. The engine creates a complex event only in cases in which the whole pattern expression is evaluated true. We consider the execution of an event pattern as an essential part of the event pattern life cycle. Therefore we described a white-box (or transparent) approach where the execution of an event pattern is monitored. We introduced a set of requirements and define a lightweight approach how an existing CEP engine could be extended with the monitoring feature. The white-box CEP is one of the main pillars since the monitoring results will be used to evolve the event patterns. Further since the next generation of real-time applications will be towards proactive computing there is a need for a transparent matching in order to gain a timely advantage by being proactive.
- **Evolution based on execution statistics:** Event patterns are subject to change. Nowadays, the evolution of patterns is done by business experts fully manually which is the usual case when business conditions changes or new requirements arise. We extended the evolution of event patterns with the concepts of execution-based evolution. Based on pattern execution monitoring we determine the execution statistics and compare current pattern execution statistics with past execution statistics. To goal was to detect deviations in the event pattern execution. This approach support the pattern engineer in keeping the event pattern repository up-to-date (continued relevance of an event pattern), to detect also possible evolution candidates the pattern engineer is not aware of automatically and hence to keep the re-development effort for an event pattern low.
- **High-level access to the underlying pattern logic:** Further we provided the PANTEON tool that implements the event pattern life cycle including the methods and approaches described above. PANTEON provides a high level graphical user interface to hide the underlying complexity. The focus is on supporting the user by keeping the barriers for event pattern management low. The tool is designed to enable non-technicians to search incrementally for the most suitable event patterns and improve continually their quality taking into account changes that might happen in the internal or external business world.

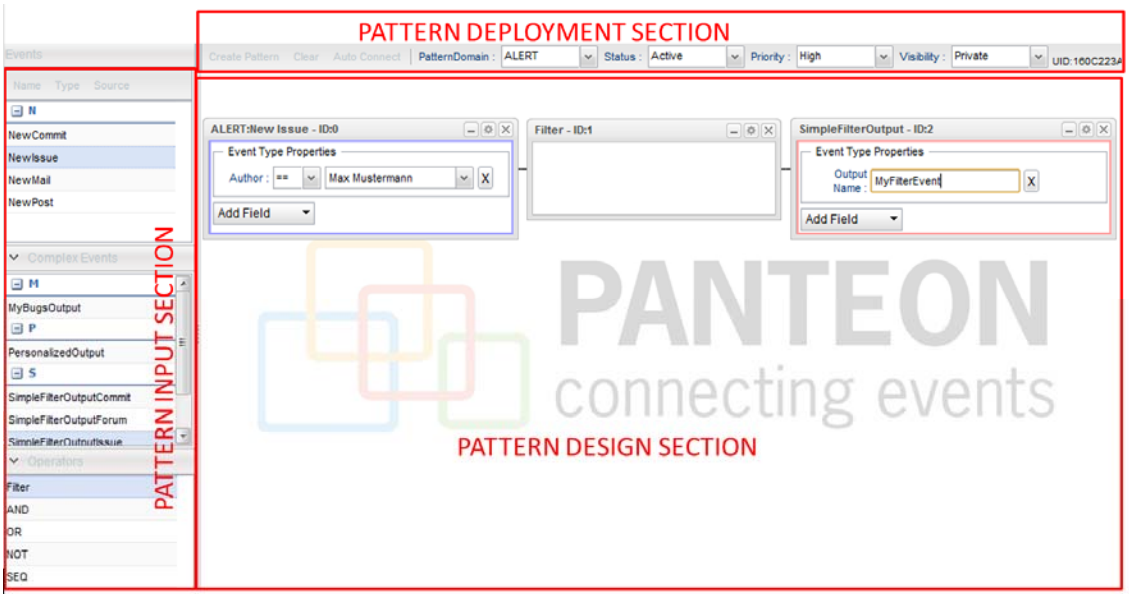

Figure 7: Interaction pattern designer

Innovation related to the real -time aspects

Today's event processing deals with the problem of detection of real time situations. Situation of interest are predefined in form of event patterns. The role of an event processing engine is to trigger a complex event as soon as an event pattern has been matched against other, previously received events. Typically it is important to trigger a complex event with as low latency as possible. This is especially true when time critical actions are triggered by detected complex events. However in some cases even complex events with low latency do not provide sufficient time for reactions. In such situations it is too late to react at the moment when a pattern is detected. Ideally, we would need a proactive reaction, that is, a reaction which is initiated before a certain complex event is recognized. For instance, instead of detecting that our SVN server has just crashed, it is more useful to detect the time when it is about to crash (e.g., when the average number of requests per second in the last 3 minutes exceeds the maximum value and the average number of requests waiting in the server does not decrease in the last 5 minutes). Our goal is to work toward detection of relevant factors (events) ahead of time so that we still have certain time to mitigate possible negative effects of the crash or to prevent it.

Although it is very useful in many applications, proactive detection of complex events is challenging in its nature. Today's Complex Event Processing (CEP) systems cannot deal with this task mainly due to the fact that they have no knowledge about situations they should discover proactively. They just detect complex events as situations that are predefined by exact event patterns. However proactive detection of complex events requires knowledge of the past (e.g., related to previous situations of interest), and comparison of situations that are detected in real time with those detected in the past. Further on, if we want that a CEP system proactively alerts us about certain situation, it needs to be capable to reason about that situation. For example, if a developing situation is similar to a previously detected, dangerous situation, or the system "knows" from the past that a dangerous situation occurs after certain events are detected, then the CEP system can proactively react. What is a dangerous situation and what is our experience about dangerous situations needs to be described in background knowledge of a particular domain of interest. Hence we notice that event processing per se is not sufficient for proactive event detection, and we advocate for an approach in which Complex Event Processing is combined with processing of domain knowledge.

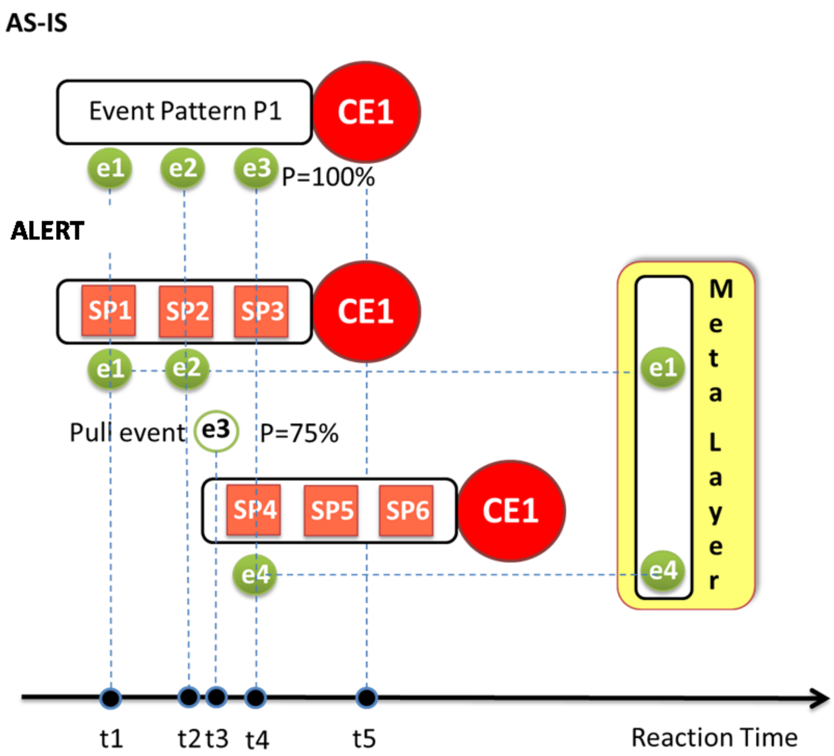

Figure 8: Monitoring of Partially Fulfilled Complex Events

The second problem that we want to address in this work is related to event pattern evolution. Namely, in today's event processing a system "blindly" tries to match event patterns against occurring events. If a pattern is not matched within a specified time window the current partial results are discarded and not processed any further. There is no analysis whatsoever which should take place in order to establish a connection between specified event patterns (situations that we would like to detect) and real event occurrences (actual happening).

In some cases certain event patterns will never be fulfilled, but information about partial fulfilment is needed to aid in pattern evolution. This information may help us in reviewing our existing patterns, and tuning them so that in future they better match the "reality".

We have created an engine that provides information about situations that are yet to be discovered. For instance, an event will be triggered whenever there is progress in detection of a certain situation. This event will carry information about the current status of the detection, and our goal is to use this status to proactively detect certain situations.

Further on, the current status of the detection of a pattern can be used for pattern statistics - to enable pattern evolution by giving information how many times a situation (complex event) has been detected, or what are parts of a complex event that has never been fulfilled. This information may be useful for pattern evolution, as it gives a hint whether a pattern is well or bad formed, and therefore it may help in evolution of patterns that can be successfully detected in the future.

We also provide a persistent storage for all events including those that provide statuses of pattern detections. This storage may be used for pattern statistics (i.e., to enable a statistical analysis, for example, how many times a situation has been detected, or what are parts of a complex event that has never been fulfilled).

1.4.2.5 *Progressing state-of the art in intelligent services*

Recommendation Service

Earlier metric-based approaches focused solely on the information found in the source code repository commit logs. For example in [11] the authors quantified the expertise of the developer in terms of Expertise Atoms (EA). In [12] the expertise of a developer for any given file is determined by creating a developer code map, which records the commit frequency of the developer correlated with the number of developers who committed a change to a particular file. In [10] the authors conducted a study where the frequency as well as the recency of a developer's activity determines his/her expertise. Finally in [13] several metrics were used in combination, such as the number and the recency of commits, the files modified, the number of lines removed or added.

The source code repository is only one of a multitude of tools that developers use; others include tools such as issue tracking systems, and communication mediums such as mailing lists, which need to be considered in order to have a holistic view of the developers' activity. In [14] eight different metrics are proposed solely from the issue tracking system which can be used to create a set of developers who can solve an issue.

These tools also contain large amounts of textual information such as commit comments, code comments, issue descriptions, issue comments, mailing list messages, etc. An alternative approach from the metric-based, is to analyze the content in this information and derive the expertise of the developer as in [15], in which the expertise of the developer is derived by analyzing the contents of the commits, for usage and implementation expertise from the API methods called and user respectively.

Recent work considers both quantitative metric-based and qualitative text-based information to determine developer expertise. In [16] expertise is determined by combining the communication score of the developer, derived from the mailing list communication history, with the development score, derived from the developer's commit frequency. In [17] expertise is modeled as an author-term-vector containing word frequencies extracted from commit diffs. Issues are modeled as a query vector containing word frequencies from the issues textual description. The expertise of a developer per issue is the cosine similarity between the vectors. In [18] the authors create a text corpus from the content in source code files and commit comments per commit. This corpus is indexed using Latent Semantic Indexing (LSI) to create a signature for each document (source code file). Using the textual description of an issue as a natural language query a ranked list of the most similar documents to an issue is returned, and from that, a ranked list of developers is derived (as in [12]). In [19] the authors use topic models to create a correlation between issues and developers who have participated in the resolution of an issue. The number of issues resolved per topic determines the expertise of the developer.

A key part of the ALERT system is suggesting to developers which issues they are most capable to solve, or to suggest developers to solve a particular issue. The user profile created in STARDOM determines the competency of the developers per class; it does not however determine the area in which the developer is competent. In order to provide more accurate suggestions we developed SOCRATES, which is a hybrid recommendation system that takes into account both the qualitative as well as the quantitative information derived from structured and unstructured information sources.

The recommendation component provides the following functionalities: suggesting issues which are most suited to a given developer and suggesting developers that are considered most capable of solving a given issue. The recommendation service has been extended to include support for a third functionality, which is to suggest the most suitable modules that developers should work on.

Two types of features are taken into account in order to provide recommendations, semantic features and competency features.

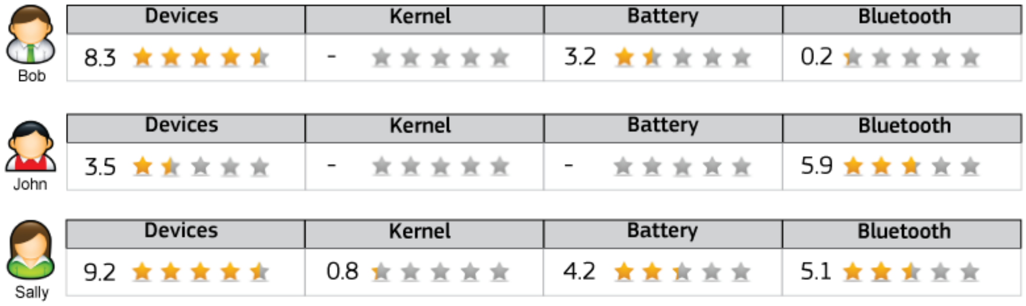

Figure 9: Developer Profiles Combining Semantic and Quantitative Features

With the term semantic features, we refer to the qualitative differences between issues and between developers. For example a new issue may refer to the use of a specialized library, which is not common throughout the project. Or, from another perspective, a developer who eventually solves this issue is also considered experienced in the use of this library. This semantic information is derived from annotations to the issues that are provided by the KEUI component.

With the term competency features, we refer to the quantitative metrics and the indices that describe the developer's previous activity in the project. These features comprise of metrics such as: the number of commits performed, the lines of code committed, the number of emails sent, etc. These metrics are extracted by the KESI component and transformed into competency indices through the STARDOM component.

The recommender takes into account both types of information, structured and unstructured, that refers to each bug and to each developer. This information is then combined in order to provide a complete developer profile. Examples of these developer profiles can be seen visualized in Figure 9.

The semantic measures of the developer profiles are transformed into vectors and a cosine similarity measure is calculated between issues and developers in order to support recommendations. The quantitative measures are used supporting the developers that have a higher competency. The computation of similarity between an issue and a developer can be seen in Eq. (1) where for developer ***d*** and issue ***i*** the similarity is calculated as a weighted sum of the cosine similarity of the semantic features and developer's competency index ***(CI)***. Similarities between developers and modules are calculated in an equivalent way.

$$sim_{d,i} = w_1 cos_{d,i} + w_2 CI_d \qquad (1)$$

The design of the proposed recommendation service is based on using characteristics of developers and open issues in order to suggest matches between them (content-based recommendation).

In a preliminary evaluation the ALERT recommendation methodology outperforms approaches that use only semantic or only quantitative features.

Visualization Service

In order for the users to be able to access the information imported from the available information sources we created an integrated user interface. The UI provides to the user a large number of functionalities that among other things include advanced search capabilities, several presentations and visualizations of search results, bug duplicate detection, recommendations, notifications and project overview.

If the user wishes to find particular information he can use the available input fields for specifying the search conditions. Beside the plain keywords he can also specify people, products, components, sources and issue id. These conditions can be added using the autosuggest feature which also makes sure there is no ambiguity in search conditions. Additionally, the results can also be limited based on time constraints and post type.

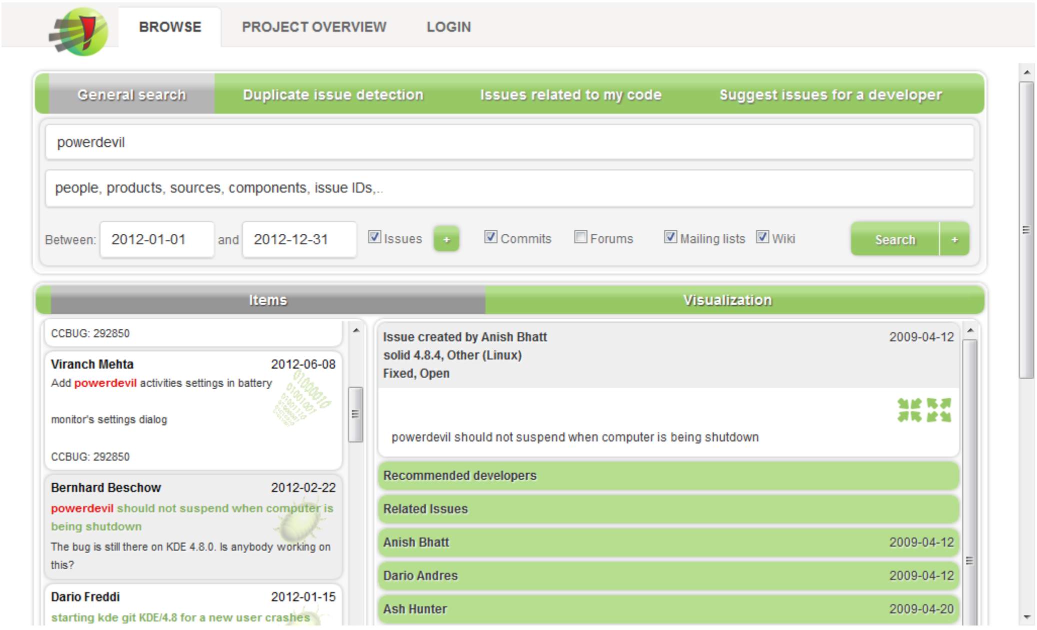

Figure 10: Search form and a list of search results

After specifying search conditions the results are presented in several ways. As common in information retrieval one way is in a form of a list of search results (see **Figure** 10). For each result we display the author of the post, the date and a short text snippet. The icon in the background indicates the type of the post (issue, code commit, forum post, email or wiki post). By clicking a result we show the detailed view of the post on the right side of the window. The displayed details depend on the type of the result. In case of the forum post or an email, for example, we display the whole thread of discussion. For code commits we display the commit description and an expandable tree view with all the files, classes and methods that were modified by the code commit. In case of an issue we display the issue meta-data (who reported it, status, resolution, OS, version, etc.), issue description and all the following issue comments. Additionally we also display the list of recommended developers who are most suitable to fix the issue.

Along with listing the results we also aggregate the results using three interactive visualizations (see Figure 11). The first visualization is a social network and displays people involved in the results. If a person A is replying to an email of the person B then the social graph displays persons A and B as nodes in the graph and there is an edge from A to B. The size of each node corresponds to the popularity of each person in the list of results – the more frequently the person is present in the results, the larger its label. Viewing the social network allows us to quickly identify the key people who are likely experts in a particular topic. If the display is cluttered with too many people we can also remove the less relevant people by clicking the minus button. Double clicking a person adds them to search conditions which allows easy refinement of search conditions.

The second visualization is a tag cloud containing the most relevant terms summarizing the list of results. This visualization allows us to obtain a shallow understanding of the topics being discussed in the results without actually reading the resulting posts. Clicking the individual terms or phrases also adds them to the search conditions.

The last visualization of search results is the timeline view. It shows the distribution of results through time. There are a number of scenarios in which this visualization is particularly useful. When searching for issues on a related topic, for example, we can identify spikes in the graph that could indicate time periods when the causes of the issues were introduced through code commits. Similarly, searching for a person can show us if the person is still active, what is his "activity pattern", etc. The visualization allows us to zoom in and also to select a time period of interest which is then set as a search condition.

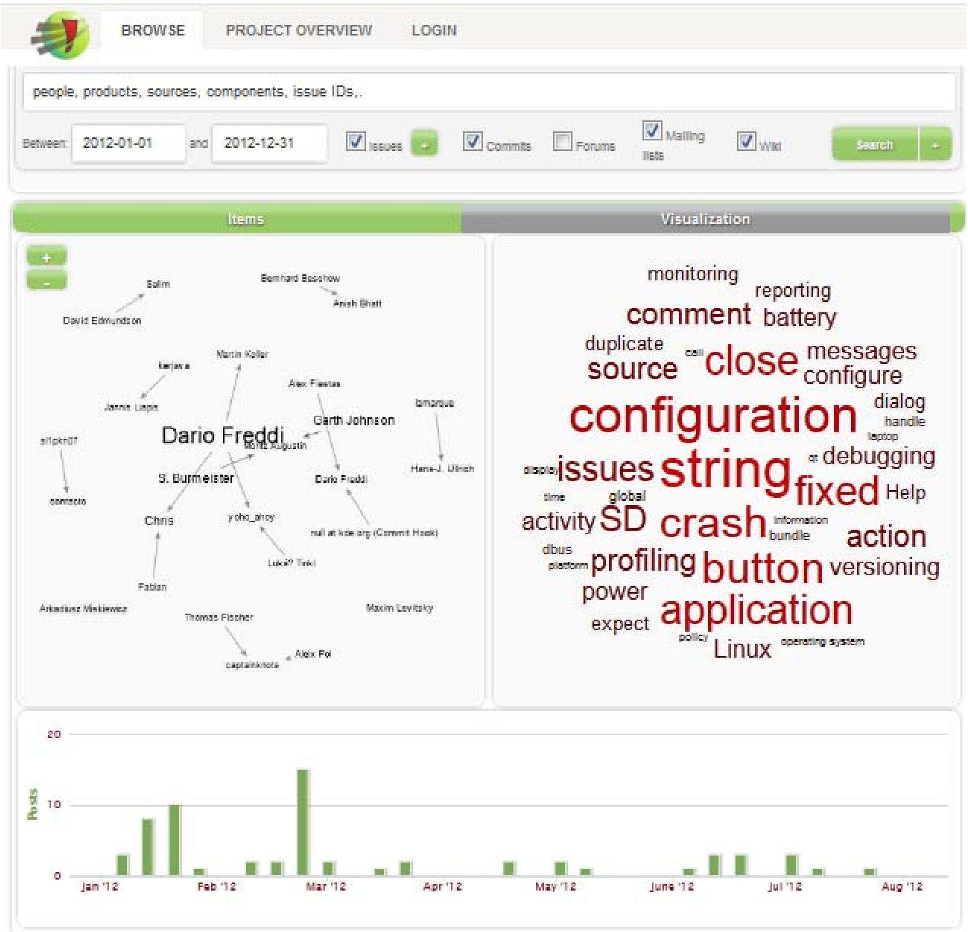

Figure 11: Aggregation of the results using three interactive visualizations

Along with visualizations of search results we developed in ALERT also two visualizations showing the overview of the whole project (see Figure 12). The visualization related to source code plots the number of commits, commiters, branches and changed files in a particular time period. Using this visualization it is easy to identify periods of high/low activity. Similarly as for the source code we also visualize information about the reported issues. For each time interval we can thus see the number of opened, closed and changed issues.

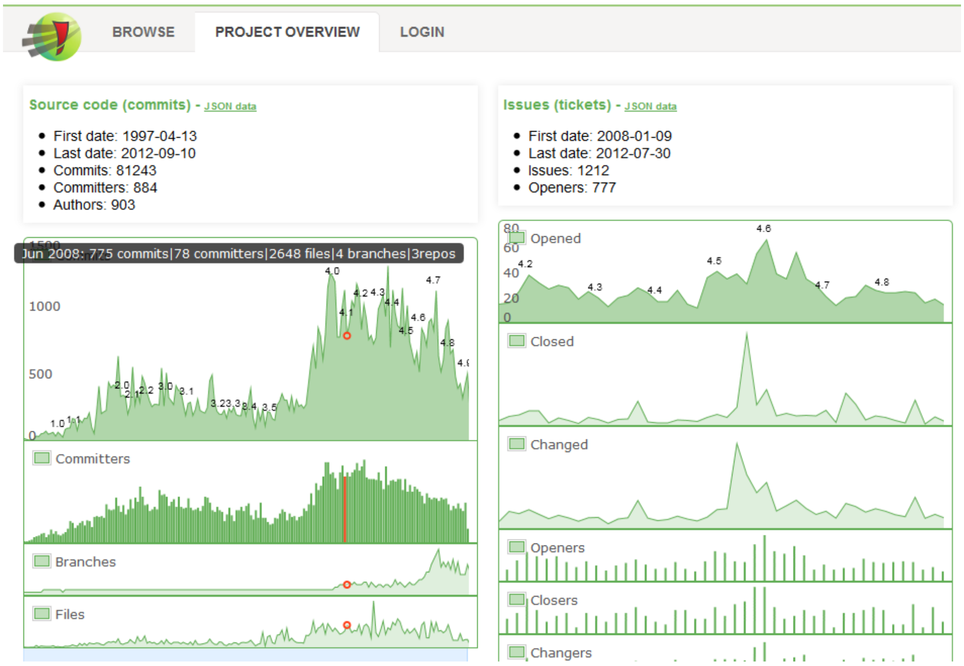

Figure 12: Overview of the project

Search Service

The Search Service of the ALERT system was written to allow external applications to access a part of ALERT functionality in a simple way. In some cases, those applications will plug directly to the ALERT event bus. However, that plugging could be relatively complex, and could expose too much functionality (such as updating of information). Therefore, a simpler, less intrusive mechanism has been provided for interconnection: a simple HTTP, RESTful API.

This HTTP API, provided by the Search Service, is intended for providing information gathered and produced by ALERT. It is targeted to external applications willing to use such information, either to show it directly to its users directly or, more commonly, to annotate some other information they are already providing. Some common cases of these external applications are:

- Scripts accessing specific information, such as those written by a developer for gathering, every morning, the information she is interested in.
- Software development forges which want to complement the information they already produce about the projects they host with some other information provided by ALERT, such as recommendations to developers.
- Integrated development environments willing to enrich the information they produce with some other provided by ALERT, such as past bug reports related to a file, or other people working in it in the past.
- Large enterprise systems which may use ALERT as a source for specialized information about software development of interest to the company.

To provide this API the Search Service works as a regular ALERT component, connecting to the ALERT event bus to access the information it needs to perform its functions.

When a query is received via the REST API, it is transformed into a request event, which is published in the ALERT bus. The ALERT system will work to answer the request, and eventually an answer event will be received. Once this happens, the answer will be used to compose the answer to the REST query. As mandated by the REST architectural style, the query will be an HTTP GET request, and the answer will be an HTTP GET response. The data provided in the response will be formated as a JSON document.

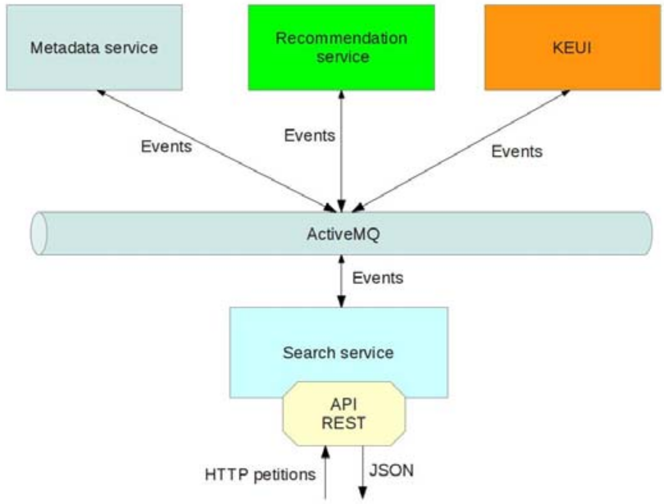

Figure 13: Relationship of the component providing the REST API (the Search Service) with the rest of the ALERT system and with external applications

The technology used for the implementation of the Search Service is based on Django/Python for the processing of data and the REST interface itself, and ActiveMQ/Python for accessing the ALERT event bus.

1.4.3 Dissemination

The dissemination task in ALERT has been a major objective during the project to reach software development, scientific and research communities. On this regard it is worth mentioning that ALERT project was present at **2012 Open World Forum** held in Paris on October 11th-13th (http://openworldforum.org), which is a leading global forum bringing together decisions-makers, developers and users from all over the world to cross-fertilize Open technological, business and societal initiatives to shape the digital future. ALERT project consortium was also present at **fOSSa Conference 2012** (https://fossa.inria.fr) held at Lille, France on December 4th, 2012. This event is considered as one of the leading global forums intended for spreading Open Source Software state of the art to academic, scientific, research communities and also to developers, early adopter and decisions-makers from all over the world. Several contacts were made on those events that have served as leads for future exploitation meetings with interested users.

During the project life a number of collateral communication materials were created such as flyers, brochures, posters, banners etc. used in several dissemination events and available via the public Website. Several other disseminations activities that can be found on ALERT final dissemination report D9.8.3 Dissemination Report.

1.5 ALERT's Results

ALERT is an open-source solution that helps developers focus on building better code. ALERT system aims at improving the coordination between developers by maintaining awareness of collaborative activities through real-time, personalized and context-aware notification.

ALERT goal is to create an active collaboration platform (i.e., a virtual actor that interacts with developers) that aims at identifying different interactions, suggesting actions and bringing past interactions into the developers' attention

enabling developers to work better together. For that, ALERT extracts information from different sources from developers' environment (mail, forums, chat, bug reports, code repository, etc.) using a set of sensors installed in the corresponding community tools. Then, it analyses the collected information extracting metadata, annotating the data and storing it using the ALERT's ontology. Different events are triggered according to the actions and interactions happened in the community. These events can be a notification, a recommendation or the execution of a service.

ALERT solution consists of:

- **A centralized ALERT service system** which is a system that holds the main architecture and functionality of all ALERT's components.
- **A set of "sensors"** that are required to proxy between ALERT system and the multiple different data sources and tools that ALERT system uses to collect information from.
- **User Application** that is the main interface where users interact with ALERT system (see Figure 10).

The **main capabilities of ALERT** that enables achieving the above goals can be summarized as follows:

- **Automation & Notifications:** to find possible duplicates or related issues and notify relevant facts to developers when they occur.
- **Live Project & Team Status**: to give more realistic picture of projects and participation with info from multiple sources.
- **Faster Team Interaction:** with Automation & Notification tools to keep developers updated.
- **Development Tools**: to integrate sensors & help coders concentrate on development instead of manual repetitive tasks.
- **Reduce Time solving Bugs**: to improve performance solving bugs and searching for possible solutions.
- **Multiple Data Sources**: to gather relevant info from multiple data sources (ITS, BTS, code repositories, blogs, forums, portals, eMail, etc.).
- **Talent Monitoring**: to improve task assignment & triage processes on OSS Communities: assign developers based on skills, expertise and participation.
- **Social Coding**: to **a**utomatically create & show developers' profiles.
- **Open Source:** ALERT is completely open-source and all its code is freely accessible on GitHub public repository.
- **Standard Ready:** ALERT comes equipped with support for the most used tools for collaboration and development (eMail, wiki sensor, GIT and SVN, etc.).

With ALERT's current capabilities we have updated our core message from what it was first stated for ALERT project, helping Open Source Communities improve their development processes and according to the studied scenarios during the project, to a more generalized approach since ALERT can help a broader range of users and more uses have been discovered that can be considered "by-products" of the project.

1.6 Who Can Benefit from ALERT?

ALERT now includes different target users and value proposition messages from its initial proposition since ALERT makes many contributions in solving problems presented in all different scenarios studied, and provides tools and methods that can help solve specific problems for users:

- ALERT is a system specially designed to improve development processes by **helping developers on debugging tasks and searching for solutions among multiple information sources**.
- ALERT can significantly **reduce the amount of time spent by developers** on solving issues, investigating and searching different sources for possible solutions with the automation and notification engine and data-collector sensors.
- ALERT can provide **analysis tools** to stay on top of the project status and team's contribution.
- And last, ALERT provides **automated profile creation** of expertise, skills and project contribution.

Therefore, ALERT software can directly help the following target users:

- **Developers and Team managers (technical and project managers)** within Software Development Companies (large organizations, SMEs and small teams as well) and Open Source Communities improve their Software Development Processes to become more productive or stay on top of project status and management information.
- **Decision makers** (technical and management areas) within Software Development Companies (large organizations, SMEs and small teams as well) and Open Source Communities that need to embrace new tools and take advantage of Social Coding trend to empower its processes and developers.
- **Developers** (novel and experts) seamlessly build a reputation with the creation of an automated skills and expertise profile while using the system.
- Additionally, ALERT technology can be used by **third party organizations** to create or extend current service offerings powered by ALERT system. This includes organizations that offer collaboration & communication tools; forge or social coding online services; desktop or online development tools; or simply professional CV profiles for job candidates.

It is important to highlight the following. First, we have included not only Open Source Communities, but also software development companies and solo developers as intended target audiences since we have found that the results and contributions obtained by ALERT research can also be applied to companies of any scale. We also include another type of target audience since it has enough potential to be attractive to third party organizations to extend these service providers portfolios to allow more interaction with developers or even allow the creation of a more interactive environment, or simply take advantage of its capability to automatically create a user profile with real contributions and experience. And second, we do not only concentrate on the message that ALERT can help them improving their development processes based on the event-driven communication, but also, we are sending two additional different value proposition messages thanks to the capability of ALERT system to automatically gather information from many different data sources and provide recommendations. Also, since ALERT can automatically create developers profiles with their skills, expertise and participation in different projects, developers can build a reputation automatically while using the system, which can also be used by companies to improve their development and talent management processes.

In order to better validate the applicability and potential impact of the results achieved during the project, ALERT technology has been tested on three real-world use cases (**OW2**, **KDE** and **Morfeo**). A two phase evaluation method has been used in order to receive valuable feed-back. In the first phase, a more informal, qualitative feedback was collected in the form of short textual comments (instant feedback). In the second phase a more detailed, quantitative evaluation questionnaire was used. Evaluation criteria and metrics have already been defined in order to receive more objective evaluation results that can clearly demonstrate and measure the impact and benefits of ALERT system. The overall result of the evaluation shows a very good acceptance of ALERT is most of the analyzed pilot functions. Feedback is especially positive on duplicate detection, extended issue view and monitoring features. The evaluation gives a high potential for the consortium in future exploitation possibilities of ALERT.

The project has also been widely disseminated in other FLOSS communities and relevant research communities; the scientific community and other research projects as well. ALERT project has also release its results under an open source business model for exploiting ALERT results after the project ends.

The expected impact is promising as there are intentions from several partners to include and use ALERT results for both, internal use and offering of new services. It is worth noting the special case of **Rey Juan Carlos University** and **KDE OSS Community**:

- Commercialization of results by creation of a spin-off company **Bitergia S.L., spin-off** from URJC, which is going to use some ALERT components to provide commercial services for FLOSS (free, libre, open source software) projects, communities, and other parties interested such as the production of standard, periodic reports; consulting and training on customized reports and specific needs; integration with forges and other software development tools; and consulting and research on specific analysis.
- The **KDE Community** will extend testing and evaluation of ALERT after the end of the project, broadening it from the initial test project (Solid) with a view to implementing ALERT for the entire family of KDE projects. KDE foresees great potential for internal use of ALERT since it is seen as "A chance to improve the way we work and make KDE software better" by allowing improvements under three categories: a) Information repository; b) Triage and self-involvement participation processes; and c) Interactive Environments:
- **All ALERT consortium members** have different interests on what ALERT will produce and due to their respective nature (industry, academic, research or community) their business goal is completely different and the benefits they expect from the project as well. **Industrial Partners** within the consortia have announce their intention to use ALERT results to offer Services around the Open Source approach of ALERT, and have shown interest on utilizing ALERT results to further test and evaluate development of new products and research lines in the future. All partners will also have another important take from the project which is they gain of knowledge and best practices, which enables them to directly sell services around gained innovations or by adopting project results and improving internal processes and operations to sell more innovative products.

1.7 Availability of results

ALERT project is fully committed to Open Source, and as such, we have released a large number of documents and software components that we want to share for free with the software development community. We have made publicly available all of these results online. In particular, all ALERT's **technical documents** and **publications** can be found at www.alert-project.eu.

Our **software components** are also available and released as open source in our website (http://www.alert-project.eu/downloads/305) along with manuals and guidelines to install and configure the system. The source code is also available on Github.com (github.com/AlertProject).

The exploitation path chosen by the project requires a **full open license approach** in all components of ALERT System which is assured since all licenses of **ALERT components are of permissive nature**: GPL v2, LGPL v3, and BSD and are clearly stated on the code on Github.com.

1.8 Potential impact of the results

After all these months working on the project we foresee great potential impact of ALERT results in different dimensions:

1.8.1 General Impact

ALERT system represents a solution that totally complies with expected impacts of the Work Programme:

- It provides new tools created out of ALERT's technology research lines that improve efficiency, responsiveness and productivity of developers by automating relevant tedious and time-consuming tasks like detection of duplicated bugs, and focusing developers effort towards those pending issues best fitted to their skills and previous expertise.
- It lowers barriers of software development companies, in particular SMEs, to develop new services by using standardized open source platforms and interfaces such as ALERT system in the context of European and global economy.
- It has the potential of creating a strengthened community of companies offering a range of different services. ALERT can be used to:
    a) help developers automatically create their own skills and expertise profile while working on regular software development processes but using empowered tools enriched by ALERT's functionality which can be used for selling project, resource or talent management services to companies; or to recruitment and job hiring agencies; and
    b) enrich current collaboration, communication and forge services and tools with interactivity and social coding capabilities.

1.8.2 Impact on Society

Some impacts of ALERT system on society and economy are:

- ALERT provides high impact on the scientific community since it has achieved substantial advancement in the state of the art in the disciplines of information extraction, including the summarization; user profiling, required for information filtering; event processing, required for information delivery; and information usage. This will enable future work and the creation of new research lines in the future.

  ALERT results are expected to be used or being utilized in the following other projects:

    - PLAY: Exchange of know-how on event processing, especially PLAY-query language which combines real-time and historical information in the complex event patterns. We organized a joint tutorial at ISWC 2012: http://tutorialiswc12.fzi.de/.
    - FITTEST projects: Point of connection through the ontology for representing bugs.

- Open Source Software Communities have proven to be tremendously beneficial to society and the economy in general in recent decades. Its impact in companies is increasing as they provide more support and trust for such initiatives. ALERT is a tool and methodology that enables and reinforces the core sense of OSS communities for the participation of its members to be done in a more efficient and productive manner.

- Therefore, ALERT will have a strong impact on this trend and the participation of large communities such KDE and OW2/Petals on the consortium has shown that it will surely impact the way they manage and use such tools to improve software development processes and the way they empower and manage members' talent in the future. It is expected that ALERT will have impact in the following aspects:
    - It will allow new ways to assess management of geographically distributed development teams providing information regarding the participation of all members in all "monitored" communication, collaboration and development tools used by the team. ALERT does not only help users but also organizations since it helps automation and collecting information from developers via installed sensors and they interactions through the UI. This will help lower the barrier since developers are always reluctant to spend time on reporting actions, because they tend to focus on what they matter most, coding.
    - It will allow what we have coined as **Focused Coding** which is benefit of using ALERT's set of tools and methods to concentrate more on the task of development avoiding spending time to perform one single task on several tools and environments when ALERT can automatically take care of that for the user behind the scenes. The hashtag **#FocusedCoding** has been used to promotion and dissemination of ALERT results on social media channels to highlight the fact that ALERT makes developers more productive by focusing on the task on only one tool and relying on automation and the intelligent decision made by the system in the background.
- We are confident that all these aspects represent high potential benefits for OSS Communities and will make them more attractive in terms of a more effective distribution of the effort among the volunteers and shorten the response time to correct failures, test new features and gather new petitions from end-users. In general, this should lead to improve the satisfaction of end-users, while at the same time its members will perceive a better participation and experience within the community, which may lead to higher commitment and increased performance.

1.8.3 Impact on the Market and Economy

The capability of ALERT system helps both sides of the development process, developers and companies, integrating and taking advantage of automation and intelligent decision systems while embracing an architecture that allows future expansion of capabilities via new sensors or new system features. This enables the possibility to customize the use the system by a large number of different types of users and companies to be capable of satisfying key societal and economical needs and trends in communication and collaboration tools; and the use of new development tools as well.

According to the studied scenarios during the project, ALERT can add value to certain market segments according to the following dimensions that can be used to position ALERT results on the market:

a) **Development Process Improvement**
b) **Collaboration & Communication Tools**
c) **Development Tools**
d) **Social Coding**

This will clearly make the advances made by ALERT reusable and expandable in the future, as the developer community evolves by adapting to new trends for those types of tools.

- The consolidation of such improvements envisaged by ALERT has also the potential to change the behaviour of businesses and the economy. Small and large organizations, not only OSS communities, can become more productive by improving development processes or by better management of projects, resources and talent within their organizations.

- Companies are in the move towards adopting Social Software tools that embrace current social and web 2.0 trends with the intention to empower collaboration and communication among their teams. Such trends also include solutions such as notifications that are gaining attention and are present in more and more tools to provide interactive and ubiquitous communications between people and between people and systems to inform about relevant and contextualized events in a personalized manner and context aware for every user. This makes business become more productive and competitive by adopting ALERT tools and methods, reducing barriers of distance among its team members and issues related to the number of development team members.
- ALERT also reduces the gap for organizations to access to disciplines like data mining, access and integration of different types of decentralized development, communication and collaborative tools, and intelligent analysis of complex data and align them to provide helpful information and tools.
- The possibility of ALERT of creating an automated profile with developers skills and expertise, if adopted not only by organizations, but by academic institutions as well, has the potential to become a very powerful asset, introducing the chance for organizations to assess real skill profiles of student seeking for a first job or by professionals searching for a new one more suited for their skills.

1.9 ALERT Consortium

Participant organization name	Part. short name	Country
FZI Research Center for Information Technology at the University of Karlsruhe	FZI	Germany
University Rey Juan Carlos - GSyC/LibreSoft group	URJC	Spain
Jozef Stefan Institute	IJS	Slovenia
Institute of Communication and Computer Systems, School of Electrical & Computer Engineering , National Technical University of Athens	ICCS	Greece
CIM Group d.o.o.	CIM	Serbia
K Desktop Environment eV	KDE	Germany
Atos Spain SA	ATOS	Spain
Corvinno Technology Transfer Center Nonprofit Ltd	CORVINNO	Hungary
Linagora SA	EBM	France

1.10 ALERT Project Contact Details

Co-ordinating partner

FZI - Forschungszentrum Informatik at the University of Karlsruhe

Haid-und-Neu-Str. 10-14

76131 Karlsruhe

Germany

Project Coordinator

Dr. Ljiljana Stojanovic

Senior Researcher / Project Manager

FZI

e-mail: Ljiljana.Stojanovic@fzi.de

Website: www.alert-project.eu

Twitter: @alertfp7

2. Use and dissemination of foreground

This section is divided in two parts: Section A which describes dissemination activities and contains public information; and Section B, which is also available as public information, contains details about the exploitable foreground and provides the plans for exploitation of ALERT project results.

2.1 Section A (PUBLIC): Dissemination

ALERT consortium has addressed dissemination task following a multidimensional approach, considering different aspects and stakeholders involved. This methodology considers the following types of dissemination activities:

a) **Project Dissemination:** the project itself has been disseminated, showing off the main facts and activity around the project such as partners involved; news related to the project among specific and relevant aspects broadcasted among several targeted communities, both academic and industrial, in order to promote the awareness of the project latest advances; news about ALERT presence, participation or organization of major events; therefore sharing the project activity itself with others to show that the project was "alive and active" by providing updated information through different online and offline channels such as the specifics sections on the website.

b) **Academic and Scientific Activities and Innovation Results and Knowledge Transfer:** these actions have focused on providing evidence of advanced scientific and research work developed by the project, compiling and sharing with the community outputs and main innovations of the project. Knowledge Transfer has been carried out covering all aspects related to transferring knowledge about research and innovations results to specific stakeholders. These actions include more specific means such as publication in technical and scientific papers, journals, conferences, webinars and demonstrations.

c) **Exploitation Marketing:** this type of dissemination has considered two aspects. First, build channels for **reaching target audiences** defined on exploitation task; and second, **awareness, branding and promotion** to send out clear and sound messages describing the project Value Proposition, Product Definition and benefits for our target audiences under two well differentiated phases. These actions have used tools such as the ALERT website, news, press releases, banners, flyers, social media presence, content seeding and collateral.

This approach has achieved the goal of giving ALERT a) **visibility** through scientific publications, and our participation in workshops, conferences and events; b) **banding quality or relevance** by its presence on relevant academic and industrial conferences, journals, and scientific media channels for scientific and research dissemination; and c) **impact of the project** through promoting the adoption of its main results and technology on different presentations, workshops and its participation in several working groups and communities in order to establish networks between interdisciplinary research institutes and companies; or with several other materials also available on the website.

This represents a summary of main dissemination results of the project:

- 2 Workshop events were organized at relevant events in the industry and scientific community, ALERT was present at **Open World forum 2012** and **fOSSa Conference 2012**. The project also attended to other 45 events & conferences (both scientific and industry oriented) at European & International level where the project was advertised and many contacts were made.

- ALERT team produced 20 Scientific Publications (papers and articles), 4 General Publications (papers and articles) were published in various relevant journals and conferences, symposia and workshops such as:
    - Empirical Software Engineering
    - International Journal of Open Source Software and Processes
    - Semantic Web Journal
    - It is worth mentioning the special case of "A Comprehensive Study of Software Forks: Reasons, Dates and Outcomes (Jesús M. Gonzalez-Barahona and Gregorio Robles from URJC)" which was selected as **Best Paper Award** at the International Conference on Open Source Systems, September 2012.
- 1 Book Chapter Article was published:
    - Semantic Complex Event Reasoning—Beyond Complex Event Processing, DOI: 10.1007/978-3-642-19797-0_14, Authors: Nenad Stojanovic, Ljiljana Stojanovic, Darko Anicic, Jun Ma, Sinan Sen, and Roland Stühmer (FZI). Foundations for the Web of Information and Services - A Review of 20 Years of Semantic Web Research. Publisher: Springer, Berlin Heidelberg, 2011, pp 253-279.

- The dissemination team produced 2 Videos of the project to use them for marketing and promotion of the project on the website and attended events and conferences,

- Collaboration with the following relevant EU projects such as PLAY among other projects such as FITTEST, HOLA! and Internet of Services Collaboration Working Group,

- The website (www.alert-project.eu) has been used as the center hub for sharing information with the community and ALERT's stakeholders, it offers and shares online access to different types of information concerning the evolution and results obtained from the project, is especially worth mentioning the availability of the software releases using a special designed online release strategy through the website: 26 blog posts and 5 newsletters, and several technical documents and guidelines were published on our website,

- The online marketing and promotion strategy of the project produced 443 tweets, 52 followers, 76 following, and 46 LinkedIn contacts.

- During the project life a number of collateral communication materials were created such as flyers, brochures, posters, banners etc. used in several dissemination events and available via the public Website. Several other disseminations activities that can be found on ALERT final dissemination report D9.8.3 Dissemination Report.

2.1.1 Data Tables

We provide next Table A1 with a list of scientific (peer reviewed) publications according article II.12 of the Grant Agreement starting with the most important ones, and another one, Table A2 with the list of all dissemination activities (publications, conferences, workshops, web sites/applications, press releases, flyers, articles published in the popular press, videos, media briefings, presentations, exhibitions, thesis, interviews, films, TV clips, posters).

TABLE A1 LIST OF SCIENTIFIC (PEER REVIEWED) PUBLICATIONS

Permanent Identifier: A permanent identifier should be a persistent link to the published version full text if open access or abstract if article is pay per view) or to the final manuscript accepted for publication (link to article in repository).

Open Access: Open Access is defined as free of charge access for anyone via Internet. Please answer "yes" if the open access to the publication is already established and also if the embargo period for open access is not yet over but you intend to establish open access afterwards.

NO.	Title	Main author	Title of the periodical or the series	Number, date or frequency	Publisher	Place of publication	Year of publication	Relevant pages	Permanent identifiers (if available)	Is/Will open access provided to this publication?
1	A Comprehensive Study of Software Forks: Reasons, Dates and Outcomes (Best paper award)	Jesús M. Gonzalez-Barahona (URJC) and Gregorio Robles (URJC)	8th International Conference on Open Source Systems	OSS 2012	Springer US	Hammamet, Tunisia	September 2012	pp 268-273		
2	Semantic Complex Event Reasoning—Beyond Complex Event Processing	Nenad Stojanovic, Ljiljana Stojanovic, Darko Anicic, Jun Ma, Sinan Sen, and Roland Stühmer (FZI)	Foundations for the Web of Information and Services - A Review of 20 Years of Semantic Web Research Heidelberg	Book Chapter	Springer Berlin Heidelberg		2011	pp 253-279	DOI: 10.1007/978-3-642-19797-0_14	
3	Real time Semantic Web: semantic technologies for real-time data processing	Guest editors: -Marko Grobelnik (JSI) -Ljiljana Stojanovic, and Nenad Stojanovic, (FZI)	Semantic Web Journal - Special issue "Special issue" for the Semantic Web Journal was accepted end of February				October 2010			

4	An approach for more efficient energy consumption based on the real-time situational awareness	Yongchun Xu, Nenad Stojanovic, Ljiljana Stojanovic and Darko Anicic (FZI)	Semantic Web Conference – ESWC	ESWC 2011		Heraklion, Greece	29/05-02/03/2011			
5	Collaboration in open source software development: the ALERT approach	Ljiljana Stojanovic, Jun Ma, Sinan Sen (FZI)	6th International Workshop on Semantic Business Process Management	ESWC 2011		Heraklion, Greece	29/05-02/03/2011			
6	ALERT: Active support and reaL-time coordination based on Event pRocessing in open source software developmenT	Ljiljana Stojanovic (FZI); Luis Cañas, Santiago Dueñas, Felipe Ortega (URJC)	15th European Conference on Software Maintenance and Reengineering	CSMR 2011	IEEE	Oldenburg, Germany	March 1, 2011,	pp 359-362		
7	ALERT project brochure/poster	Dusan Zirojevic (CIM)	Career Days 2011				19/04/2011			
8	An Intelligent Event-driven Approach for Efficient Energy Consumption in Commercial Buildings: Smart Office Use Case.	Authors: Nenad Stojanovic, Dejan Milenovic, Yongchun Xu, Ljiljana Stojanovic, Darko Anicic, Rudi Studer (FZI)	5th ACM International Conference on Distributed Event-Based Systems (DEBS)	DEBS 2011		New York (USA)	11-15/ 07/2011			
9	ALERT: semantic event-driven collaborative platform for software development	Ljiljana Stojanovic (FZI), Sinan Sen (FZI), Jun Ma (FZI), Dominik Riemer (FZI)	Proceedings of the Sixth ACM International Conference on Distributed Event-Based Systems	DEBS 2012		Berlin, Germany	July 16-20, 2012			

10	Using Complex Event Processing for Modeling Semantic Requests in Real-Time Social Media Monitoring	Dominik Riemer (FZI), Ljiljana Stojanovic (FZI), Nenad Stojanovic (FZI)	Proceedings of the 1st International Workshop on Real-Time Analysis and Mining of Social Streams (RAMSS 2012) in conjunction with the International Conference on Weblogs and Social Media (ICWSM).	RAMSS 2012		Dublin, Ireland	2012			
11	Using Complex Event Processing for Modeling Semantic Requests in Real-Time Social Media Monitoring. Are Developers Fixing their Own Bugs?: Tracing Bug-Fixing and Bug-Seeding Committers	Daniel Izquierdo-Cortázar (URJC), Andrea Capiluppi and Jesús M. González-Barahona (URJC)	International Journal of Open Source Software and Processes	Volume 3, Issue 2.			November 2011	20 pages.	DOI: 10.4018/jossp.2011040102 http://www.igi-global.com/article/developers-fixing-their-own-bugs/62098	
12	Modification and Developer Metrics at the Function Level: Metrics for the Study of the Evolution of a Software Project	Gregorio Robles (URJC), Israel Herraiz, Daniel M. German, and Daniel Izquierdo-Cortazar (URJC)	3rd International Workshop on Emerging Trends in Software Metrics (WETSoM 2012)	WETSoM 2012	IEEE	Zurich, Switzerland	3 June 2012	pp 49-55	http://agile.diee.unica.it/wetsom2012/program.html	
13	Effort estimation of FLOSS projects: a study of the Linux kernel	Andrea Capiluppi and Daniel Izquierdo-Cortázar (URJC)	Empirical Software Engineering	Volume 18, Issue 1 ,	Springer US		December 2011	pp 60-88	DOI: 10.1007/s10664-011-9191-7 http://www.springerlink.com/conte	

									nt/612r616k 8t52m867/	
14	On the reproducibility of empirical software engineering studies based on data retrieved from development repositories	Jesús M. González-Barahona (URJC) and Gregorio Robles (URJC)	Empirical Software Engineering	Volume 17, Issue 1-2	Springer US		February 2012	pp 75-89,	DOI: 10.1007/s10664-011-9181-9 http://www.springerlink.com/content/j501w21183375u1r/	
15	Do more experienced developers introduce less bugs?	Daniel Izquierdo-Cortázar (URJC), Jesús M. Gonzalez-Barahona (URJC) and Gregorio Robles (URJC)	8th International Conference on Open Source Systems	OSS 2012	Springer US	Hammamet, Tunisia	September 2012	pp 1-14	http://oss2012.org/program#detailed	
16	Combining Activity Metrics and Contribution Topics for Software Recommendations	Konstantinos Christidis, Fotis Paraskevopoulos, Dimitris Panagiotou and Gregoris Mentzas from ICCS	Third International Workshop on Recommendation Systems for Software Engineering,	RSSE 2012		Zurich, Switzerland	June 4 2012			
17	Enabling Semantic Search in Large Open Source Communities	Gregor Leban, Lorand Dali, Inna Novalija (JSI)	9th Extended Semantic Web Conference - ESWC 2012	ESWC 2012		Hraklon, Greece	July 4th 2012			

18	Searching for information in software development projects using ALERT system	Luka Stopar, Gregor Leban (JSI)	15th International multiconference	IS 2012		Jožef Stefan Institute, Jamova cesta 39, Ljubljana, Slovenia	8 - 12 October, 2012			
19	Analysis and prediction of bug duplicates in KDE bug tracking system	Gregor Leban (JSI)	14th International multiconference	IS 2011		Jožef Stefan Institute, Jamova cesta 39, Ljubljana, Slovenia	10 - 14 October 2011, 2011			
20	Searching for information in software development projects using the ALERT system	Luka Stopar, Gregor Leban, (JSI)	Information society				October 2012			
21	A Methodology for Designing Events and Patterns in Fast Data Processing	Dominik Riemer, Nenad Stojanovic, Ljiljana Stojanovic, (FZI)	CAiSE Conference 2013				2013			

Table 2: List of scientific (peer reviewed) publications, starting with the most important ones

TABLE A2 LIST OF DISSEMINATION ACTIVITIES

Type of activities: publications, conferences, workshops, web, press releases, flyers, articles published in the popular press, videos, media briefings, presentations, exhibitions, thesis, interviews, films, TV clips, posters, Other

Type of audience: Scientific Community (higher education, Research), Industry, Civil Society, Policy makers, Medias, Other (multiple choices is possible)

NO.	Type of activities	Main Leader	Title	Date/Period	Place	Type of Audience	Countries Addressed
1	Presentation	URJC - Presentation of Jesus Barahona in Best Practices for FLOSS working group.	EC Collaboration Meeting Organized by the Internet of Services unit of ICT program of European Commission.	19-20/10/ 2010	Brussels	Industrial and Scientific Community	European
2	Presentation	FZI - Presentation of Ljiljana Stojanovic in Semantics working group about the role of semantics in ALERT	EC Collaboration Meeting Organized by the Internet of Services unit of ICT program of European Commission.	19-20/10/ 2010	Brussels	Industrial and Scientific Community	European
3	Conference	ATOS - Submitted presentation in the Use Cases working group about the Morfeo use case in ALERT Note: Clara Pezuela (Atos) was planned to represent ALERT in this event, but due to flight troubles she couldn't attend, although the presentation was sent to the group	EC Collaboration Meeting Organized by the Internet of Services unit of ICT program of European Commission.	19-20/10/ 2010	Brussels	Industrial and Scientific Community	European

4	Conference	FZI - Marko Grobelnik (Josef Stefan Institute, Slovenia-SL) Nenad Stojanovic (FZI Research Center for Information Technologies, Karlsruhe, Germany-DE) Ljiljana Stojanovic (FZI Research Center for Information Technologies Karlsruhe, Germany-DE)	Semantic technologies for real-time data processing: state of the art, application opportunities and challenges. Extended Semantic Web Conference – ESWC 2011 (Tutorial)	02/06/2011	Greece	Scientific Community	European
5	Conference	FZI - Nenad Stojanovic (FZI) & Ljiljana Stojanovic (FZI)	6th International Workshop on Semantic Business Process Management collocated with ESWC Extended Semantic Web Conference 2011. http://sbpm2011.fzi.de	29/05/2011 02/06/2011	Greece	Scientific Community	European
6	Conference	Corvinno - Attendee András Gábor	Future Internet Day	09/12/2010	Budapest (H)	Industrial and Scientific Community	National
7	Presentation	FZI - Tutorial: Grand Challenge: The Global Event Processing Fabric and its Applications. Authors: Nenad Stojanovic, Pedro Bizzaro and K. Mani Chandy Efficient energy consumption in a smart office based on intelligent complex event processing. Authors: Yongchun Xu, Ljiljana Stojanovic, Jun Ma, Darko Anicic and Nenad Stojanovic Demo:	DEBS 2011: 5th ACM International Conference on Distributed Event-Based Systems (DEBS)	11-15/ 07/2011	New York (USA)	Industrial and Scientific Community	International

		Complex Event Pattern Evolution based on Real-Time Pattern Execution Statistics. Authors: Sinan Sen, Ruofeng Lin and Bijan Fahimi Shemrani					
8	Presentation	Corvinno - Moderate a workshop -Péter Fehérc	ECKM	1-2/09/2011	Passau (D)	Industrial and Scientific Community	international
9	Presentation	Corvinno - Act as speaker András Gábor	SKIMA (supported by IEEE)	8-11/09/ 2011	Benevento (IT)	Industrial and Scientific Community	international
10	Presentation	JSI - Promoting ALERT with flyers – Gregor Leban	WWW conference	28/03-01/04/2011	Hyderabad, India	Industrial & Scientific Community	International
11	Presentation	CIM - Speaker, panel - Dusan Zirojevic	Career Days 2011	19/04/2011	Nis, Serbia	Industrial, Students and Scientific Community	national
12	Conference	FZI - Organisation & presentation	FZI in Dialog, Topic "Complex Event Processing	24/11/2010	Karlsruhe, Germany	Industrial	National

13	Conference	FZI - Organization of the competition for event-driven applications	DEBS 2011 Event Processing Challenge	13/07/2011	NY, USA	Industrial and Scientific Community	International
14	Presentation	FZI - Organisation	5th International Workshop on Event-Driven Business Process Management accepted to BPM conference	28/08/2011 02/09/2011	Clermont-Ferrand, France	Industrial and Scientific Community	International
15	Presentation	FZI - Presentation: ALERT in the “Real-time data management and processing” CWG.	Semantics in Services and Clouds Session: Internet of Services Collaboration Meeting	28-29/09/2011	Brussels, Belgium	Scientific Community	European
16	Presentation	FZI - Presentation	3rd WOSS - Workshop on Software Services	19-21/09/2011	Szczecin Poland	Scientific Community	European
17	Conference	FZI - Lectures	senZations summer school	29/08/ 2011 02/09/2011	Kotor, Montenegro	Scientific Community	European
18	Conference	KDE - Attendance - Stuart Jarvis	Desktop Summit https://desktopsummit.org /	6-12/08/2011	Berlin, Germany	Industrial and academic	Europe

19	Conference	KDE - Attendance	FOSDEM http://fosdem.org/2011/	4-5/02/2011	Brussels, Belgium	Developers	Europe
20	Presentation	ATOS & FZI - Ljiljana Stojanovic (FZI) Submitted and presented the ALERT use cases, use of S-Cube methodology – Presentation by Clara Pezuela -Poster submission -Flyer distribution	EC Collaboration Meeting Organized this year by HOLA! Project and by the Internet of Services unit of ICT program of European Commission.	27-28/09/2011	Brussels, Belgium	Industrial and Scientific Community	European
21	Presentation	Clara Pezuela (ATOS)	fOSSa 2011	26/10/2011	Lyon (France)	OS companies, Scientific Community	Europe
22	Presentation	Alejandra Trujillo (ATOS)	RCIS (Industrial Track)	18/05/2012	Valencia (Spain)	Software Industry and Scientific Community	Europe
23	Presentation	(URJC)	Workshop on metrics, evaluation and management of open online communities	July 9-10, 2012		Industry and Scientific Community, attendance by direct invitation	International5

24	Presentation	(URJC)	Wikimania 2012	July 11-15, 2012		Industry and Scientific Community, entrepreneurs and practitioners	International38
25	Presentation	(URJC)	Mining Summer School MSR Vision 2020	August 20-24, 2012		Scientific Community, Industry	International10
26	Presentation	(ICCS)	3rd International Workshop on Recommendation Systems for Software Engineering. In conjunction with ICSE 2012	4 June 2012	Zurich, Switzerland.	Software engineering scientists and practitioners in the area of Recommendation Systems for Software Engineering. Also attendees of the ICSE, a large international conference on Software Engineering sponsored by ACM and IEEE.	International
27	Conference	(ICCS)	6th ACM Conference on Recommender Systems 2012	9-13/09/2012		Scientific Community (Researchers and Practitioners in the area of Recommender Systems)	International

28	Conference	(CIM)	“Project Proposal Preparation”, within the International Programme, AID FOR TRADE“	05-07/10/2011		Industrial	Serbia
29	Conference	(CIM)	Danub.IT – Danubeshoring Business & IT	20-21/10/2011		Industrial and Scientific Community	International
30	Conference	(CIM)	Information Day for FP7 ICT Call 8 and Call 9	07/12/2011		Industrial and Scientific Community	Serbia
31	Conference	(CIM)	International Information Technology Show - Workshop at Computex, Taipei, Taiwan.	05.06 – 09.06.2012	Taipei, Taiwan	Industrial and Scientific Community	International
32	Conference	(KFT)	5th International Conference on Integrated Systems: Design and Technologies	12-15 May 2012	Mallorca (Spain)	Industrial and Scientific Community	EuropeEU
33	Conference	(KFT)	Conference of the Hungarian Economic Society	41181	Hungary	Industrial and Scientific Community	EuropeHU

34	Workshop	Ljiljana Stojanovic and Dominik Riemer (FZI); Stuart Jarvis and Dario Freddi (KDE); Jesús González-Barahona (URJC); Clara Pezuela and Oliver Barreto (ATOS); Gregor Leban (JSI); Fotis Paraskevopoulos (ICCS)	Open World Forum 2012 ALERT workshop	12 Oct 2012,	Paris, France	Open Source Developers, Industry and Scientific Community	International
35	Workshop	Gregor Leban (JSI); Fotis Paraskevopoulos (ICCS)	fOSSa Conference 2012 ALERT workshop	4 December, 2012	Lille, France	Open Source Developers, Scientific Community	International
36	Conference	Gregor Leban (JSI)	Internet of Services, Open source supporting technologies	16 October, 2012		Scientific Community	International
37	Conference	Gregor Leban (JSI)	Internet of Services, Semantics in services	17 October, 2012		Scientific Community	International
38	Conference	András Gábor (Corvinno)	Scientific and Education Forum for Business Information System	9-10/11/2012		Scientific Community, Industry	International

39	Conference	Zoltán Szabó (Corvinno)	Corvinus University Research Day	09/11/2012		Scientific Community, Industry	Hungary
40	Conference	Peter Wolf (Corvinno)	Territorial LivingLab International Seminar	5-6/2/2013		Scientific Community, Industry	EU
41	Conference	URJC	FOSDEM 2013	03/02/2013	Brussels, Belgium	Open Source developers and users	International
42	Presentation	Ljiljana Stojanovic, Nenad Stojanovic, (FZI)	Tutorial "Scalable semantic processing of huge, distributed real-time streams" at ISWC 2012	12/11/2012		Scientific Community, Industry	World-wide
43	Web	Balazs Cseh (Corvinno)	Deployment & evaluation plan http://alert-project.eu/content/deployment-evaluation-plan	21 July 2011		Scientific Community	International
44	Web	Sisan Sen (FZI)	ALERT Conceptual Architecture http://alert-project.eu/content/alert-conceptual-architecture	18 July 2011		Scientific Community	International

45	Web	Dušan Zirojević (CIM)	Metadata service http://alert-project.eu/content/metadata-service	14 July 2011		Scientific Community	International
46	Web	Ignacio Cañas (URJC)	Improvements in the extraction of issues from ITS/BTS http://alert-project.eu/content/improvements-extraction-issues-itsbts	9 June 2011		Scientific Community	International
47	Web	Alejandra Trujillo (ATOS)	Open Source Communities http://alert-project.eu/content/open-source-communities	19 April 2011		Scientific Community	International
48	Web	Laurent Lacote (Linagora)	Functional requirements: Going from dreaming to realizing http://alert-project.eu/content/functional-requirements-going-dreaming-realizing	14 March 2011		Scientific Community	International
49	Web	Mauricio Ciprian (ATOS)	A brief view of ALERT architecture http://alert-project.eu/content/brief-view-alert-architecture	10 March 2011		Scientific Community	International
50	Web	Javier Rodriguez (ATOS)	Welcome to Alert Project's Blog http://www.alert-project.eu/content/welcome-alert-projects-blog	3 March 2011		Scientific Community	International

51	Web	Dušan Zirojević (CIM)	Using NLP and Linked Open Data Repositories for Ontology Learning: The OCELOt approach http://www.alert-project.eu/content/technical-evaluation-alert-integrated-system	September 4th 2012		Scientific Community	International
52	Web	Felipe Ortega (URJC)	Information extraction from version control and bug trackers http://www.alert-project.eu/content/information-extraction-version-control-and-bugtrackers	August 2nd 2012		Scientific Community	International
53	Web	Laurent Lacote (EBM)	ALERT, one tool to bring them all together http://www.alert-project.eu/content/alert-one-tool-bring-them-all-together	June 14th, 2012		Scientific Community	International
54	Web	Mauricio Ciprian (ATOS)	Using NLP and Linked Open Data Repositories for Ontology Learning: The OCELOt approach. http://www.alert-project.eu/content/using-nlp-and-linked-open-data-repositories-ontology-learning-ocelot-approach	May 14th 2012		Scientific Community	International
55	Web	Gregor Leban (IJS)	How to automatically detect bug duplicates http://www.alert-project.eu/content/how-automatically-detect-bug-duplicates	March 20th 2012		Scientific Community	International

56	Web	Stuart Jarvis (KDE)	Working With the ALERT Partners http://www.alert-project.eu/content/working-alert-partners	February 2nd 2012		Scientific Community	International
57	Web	Dimitris Panagiotou (ICCS)	ALERT Ontologies http://www.alert-project.eu/content/alert-ontologies	October 19th 2012		Scientific Community	International
58	Web	Stuart Jarvis (KDE)	Testing ALERT in KDE http://www.alert-project.eu/content/testing-alert-kde	March 1st, 2013		Scientific Community	International
59	Web	(ICCS)	Topic based developer expertise http://www.alert-project.eu/content/topic-based-developer-expertise	January 17th 2013		Scientific Community	International
60	Web	Balazs Cseh (Corvinno)	Trials have been started http://www.alert-project.eu/content/trials-have-been-started	December 14th 2012		Scientific Community	International
61	Web	Doiminik Riemer (FZI)	Pushing Wiki Articles to the ALERT System http://www.alert-project.eu/content/pushing-wiki-articles-alert-system	November 19th, 2012		Scientific Community	International

62	Web	Saša Stojanović, (CIM)	The second release of the ALERT system http://sasa-stojanovic.blogspot.com/2013/03/2nd-release-of-alert-system-is-ready.html	March 19th, 2013		Scientific Community	International
63	Web	Stuart Jarvis, KDE (personal blog)	“Some final steps for ALERT”	13 March 2013		Scientific Community	International
64	Web	Stuart Jarvis, KDE (Alert Project blog)	“Testing ALERT in KDE”	1 March 2013		Scientific Community	International
65	Web	Stuart Jarvis, KDE (personal blog)	“Presenting ALERT at OWF”	19 October 2012		Scientific Community	International
66	Web	Stuart Jarvis, KDE (personal blog)	“ALERT UI Demo (Part 1: Overview)”	13 November 2012		Scientific Community	International
67	Web	Stuart Jarvis, KDE (personal blog)	“ALERT meeting in Paris – and OWF Workshop”	10 October 2012		Scientific Community	International

68	Web	Stuart Jarvis, KDE (personal blog)	"ALERT Project 4th newsletter available"	2 October 2012		Scientific Community	International
69	Publication	Stuart Jarvis (KDE)	KDE e.V. COMMUNITY REPORT. Title: "The ALERT Project". 1 ST QUARTER, 2012 | ISSUE 21 Published online: http://ev.kde.org/reports/ev-quarterly-2012_Q1.pdf	26/06/2012		Developers, Industrial and Scientific Community	International
70	Presentation	CIM (Dusan Zirojevic), FZI (Sinan Sen, Darko Anicic, Jun Ma, Dominik Riemer, Ljiljana Stojanovic), IJS (Gregor Leban), ICCS (Dimitris Panagiotou, Fotis Paraskevopoulos)	Danub.IT – Danubeshoring Business & IT – ALERT Project Danub.IT – Danubeshoring Business & IT	20-21/10/2011		Developers, Industrial and Scientific Community	International
71	Presentation	CorvinnoKFT	Development for developers Innoteka	August 2012		Developers, Industrial and Scientific Community	International
72	Poster	ALERT Consortium	Project material (poster and leaflet) was distributed during the Concertation Meeting 2011.	2011		Industrial and Scientific Community	International
73	Flyers	ALERT Consortium	Project material (poster and leaflet) was distributed during the Concertation Meeting 2011.	2011		Industrial and Scientific Community	International

74	Poster	ALERT Consortium	ALERT poster was showed at the Digital Agenda Assembly http://ec.europa.eu/information_society/digital-agenda/daa/index_en.htm on June 16th and 17th, 2011 in Brussels by request of the Project Officer	2011	Brussels	Industrial and Scientific Community	International
75	Web	KDE	Blog post from KDE Community developers: http://euroquis.nl/bobulate/?p=1197	2011		Developers, Industrial and Scientific Community	International
76	Web	KDE	Blog post from KDE Community developers: http://euroquis.nl/bobulate/?p=1204	2011		Developers, Industrial and Scientific Community	International
77	Web	KDE	Blog post from KDE Community developers: http://www.asinen.org/2011/05/some-news-from-the-alert-project/	2011		Developers, Industrial and Scientific Community	International
78	Web	ATOS	ALERT's project results and ongoing achievements were also put on the website. We have included the link to our newsletter and blog http://www.it-tude.com/news.html.	2011		Industrial and Scientific Community	International

79	Collaboration	ALERT Consortium	ALERT has become an associate project at IT-TUDE http://www.it-tude.com/alert.html	2011		Scientific Community	International
80	Web	KDE	Blog post from KDE Community developers:	2012		Developers, Industrial and Scientific Community	International
81	Web	KDE	Blog post from KDE Community developers:	2012		Developers, Industrial and Scientific Community	International
82	Web	KDE	Blog post from KDE Community developers:	2012		Developers, Industrial and Scientific Community	International
83	Web	KDE	Blog post from KDE Community developers:	2012		Developers, Industrial and Scientific Community	International
84	Web	KDE	Blog post from KDE Community developers:	2012		Developers, Industrial and Scientific Community	International

85	Web	KDE	Blog post from KDE Community developers:	2011		Developers, Industrial and Scientific Community	International
86	Web	KDE	Blog post from KDE Community developers:	2011		Developers, Industrial and Scientific Community	International
87	Web	CIM	Web page on CIM Group ltd. web site dedicated to promotion of ALERT Project, containing description of the Project (in English and Serbian), ALERT video and newsletter available for download - http://www.cimgrupa.eu/sr/ShowArticle.aspx?ID=240			Industrial and Scientific Community	International
88	eMail Campaign	CIM	Distribution of ALERT newsletters to relevant institutions from the Balkan region selected from the CIM's contacts database -			Industrial and Scientific Community	Balkan region
89	Press Release	KFT	Improvements for developers Article (Fejlesztések fejlesztők számára) -			Industrial and Scientific Community	International

90	Web	ATOS	IT-Tude - ALERT's project results and ongoing achievements were also put on the website. We have included the link to our newsletter and blog http://www.it-tude.com/news.html.			Developers, Industrial and Scientific Community	International
91	Web	ATOS	IT-Tude - ALERT has become an associate project at IT-TUDE http://www.it-tude.com/alert.html.			Developers, Industrial and Scientific Community	International
92	Web	ATOS	IT-Tude - We have included our logo on IT-tude's web site and included IT-tude's logo in ALERT web site.			Developers, Industrial and Scientific Community	International
93	Presentation	Gregor Leban (JSI)	Presentation on Annotation ontology construction and duplicate detection at FZI -	02/04/2012	Karlsruhe, Germany.	Scientific community	Europe
94	Presentation	Gregor Leban (JSI)	Presentation on ALERT functionality at JSI -	09/05/2012	Slovenia	Scientific community	Europe

95	Publication	CIM	ALERT flyer translated into Serbia, Bosnian, Croatian and Macedonian Languages -			Scientific community	Balkan region
96	eMail Campaign	CIM	Distribution of ALERT newsletters and translated materials to relevant institutions from the Balkan region selected from the CIM's contacts database -			Scientific community	Balkan region
97	Web	CIM	ALERT banners were posted via CIM's Twitter Account - https://twitter.com/cimgroupsrs			Scientific community	International
98	Web	CIM	ALERT project news was disseminated via CIM's web site (within the "News" section as well as on the page dedicated to ALERT project where a link towards ALERT's newsletters has been placed). - http://www.cimgrupa.eu/sr/ShowArticle.aspx?ID=240			Scientific community	International
99	Web	Alejandra Trujillo & Oliver Barreto (ATOS)	Content and design updates - http://www.alert-project.eu			Developers, Industrial and Scientific Community	International

100	Web	LINAGORA	ALERT now available to trial! - https://research.linagora.com/display/alert/2013-02-01+ALERT+now+available+to+trial!	01/02/2013		Developers, Industrial and Scientific Community	International
101	Web	LINAGORA	ALERT manuals publicly available - https://research.linagora.com/display/alert/2013-03-08+ALERT+manuals+publicly+available	08/03/2013		Developers, Industrial and Scientific Community	International
102	Web	LINAGORA	VM update, to integrate latest components! - https://research.linagora.com/display/alert/2013-03-20+VM+update%2C+to+integrate+latest+components!	20/03/2013		Developers, Industrial and Scientific Community	International
103	Collaboration Meeting	FZI	MOSAIC (STREP) - MOSAIC could integrate ALERT tools for improving bug resolution into their platform - Meeting in FIA Budapest (Dana Petcu, coordinator of Mosaic and Nenad Stojanovic, technical coordinator of ALERT)	2011	Budapest, Humgary	Scientific Community	Europe
104	Workshop		MOSAIC (STREP) - - Participation of a workshop in WoSS	2011		Scientific Community	Europe

105	Collaboration Meeting		MOSAIC (STREP) - - Specific collaboration for the API development	2011		Scientific Community	Europe
106	Web	ATOS	MOSAIC (STREP) - - Reference site on our web site	2011		Scientific Community	Europe
107	Collaboration Meeting	ATOS & FZI	MOSAIC (STREP) - - Meeting in Concertation Meeting 2011	2011	Brussels, Belgium	Scientific Community	Europe
108	Web	ATOS	HOLA! - - Registration in Hola! Portal to take advantage of the dissemination means they will offer	2011		Scientific Community	Europe
109	Workshop	ATOS	HOLA! - - Attendance to webinar organized via Hola! Portal about Open Source	2011		Scientific Community	Europe
110	Web	ATOS	HOLA! - - Reference site on our web site	2011		Scientific Community	Europe

111	Collaboration Meeting	ATOS	HOLA! - - Meeting in Concertation Meeting 2011	2011		Scientific Community	Europe
112	Collaboration Meeting	ATOS & FZI	S-Cube - Use cases methodology - Collaboration Meeting for Utilization of the methodology proposed by S-Cube project to define ALERT use cases with some adaptations	2011		Scientific Community	Europe
113	Web	ATOS	S-Cube - - Publication of ALERT use cases in S-Cube repository: http://scube-casestudies.ws.dei.polimi.it/index.php/Main_Page	2011		Scientific Community	Europe
114	Workshop	ATOS & FZI	S-Cube - - Participation in Collecting Use Cases working group session to present ALERT use cases and the experience of using the methodology	2011		Scientific Community	Europe
115	Web	ATOS	S-Cube - - Reference site on our web site	2011		Scientific Community	Europe

116	Collaboration Meeting	ATOS & FZI	SPRERS - Dissemination - SPRERS will help us in participating and increasing ALERT collaboration activities with other projects through thematic workshops, expert meetings, a training event that are organised by this project. Also in increasing the awareness of the project among the ICT community	2011		Scientific Community	Europe
117	Presentation		SPRERS - Collaboration, participation in events, find synergies with other projects through meetings, etc - Invited talk in WoSS (by Sinan Sen)	2011		Scientific Community	Europe
118	Web	ATOS	SPRERS - - Reference site on our web site	2011		Scientific Community	Europe
119	Collaboration Meeting	Clara Pezuela (ATOS)	FITTEST (STREP) - Meeting by phone in July 2011 (Tanja Vos, coordinator of Fittest and Clara Pezuela, dissemination responsible of ALERT). Exchange of documentation between two projects. Point of connection through the ontology for representing bugs - Information extraction, the output of testing is sometimes bug management, ontology	July 2011		Scientific Community (higher education, Research)	Europe

120	Web	Clara Pezuela (ATOS)	FITTEST (STREP) - Reference site on our web site -	2011		Scientific Community (higher education, Research)	Europe
121	Collaboration Meeting	Clara Pezuela (ATOS)	FITTEST (STREP) - Meeting in Concertation Meeting 2011 -	2011	Meeting in Concertation Meeting 2011	Scientific Community (higher education, Research)	Europe
122	Workshop	Clara Pezuela (ATOS)	FITTEST (STREP) - Joint publication and workshop at RCIS 2012 -	2012	Joint publication and workshop at RCIS 2012	Scientific Community (higher education, Research)	Europe
123	Web	Clara Pezuela (ATOS)	PLAY (STREP) - Reference site on our web site -	2012		Scientific Community (higher education, Research)	Europe
124	Workshop	Clara Pezuela (ATOS)	PLAY (STREP) - Joint workshop at Extended Semantic Web conference 2012 -	2012		Scientific Community (higher education, Research)	Europe
125	Workshop	Clara Pezuela (ATOS)	PLAY (STREP) - Workshop: Scalable semantic processing of huge, distributed real-time streams -	2012		Scientific Community (higher education, Research)	Europe

126	Workshop	Clara Pezuela (ATOS)	PLAY (STREP) - Workshop: Semantic Business Process Management -	2012		Scientific Community (higher education, Research)	Europe
127	Collaboration Meeting	Clara Pezuela (ATOS)	HOLA! - Updating of the news/workshop/deliverables/video -			Scientific Community (higher education, Research)	Europe
128	Collaboration Meeting	Clara Pezuela (ATOS)	HOLA! - Poll about portal benefits for the project and usability -			Scientific Community (higher education, Research)	Europe
129	Collaboration Meeting	Ljiljana Stojanovic (FZI)	ARtTSENSE - Meeting: ARtSENSE follows the ALERT approach for integration of heterogeneous sensor data using semantically enriched events and the ActiveMQ as a middleware. On the other hand, the initial ALERT metadata service was developed by considering the corresponding ARtSENSE component. - Sensor Integration, Semantics	2012		Scientific Community (higher education, Research)	Europe

130	Collaboration Meeting	Ljiljana Stojanovic (FZI)	ReFLEX - Hybrid complex-event detection engine developed within the ReFLEX projects as well as concept of generic adaptors for arbitrary (and previously un-known) sensors could be very useful for the further extension of the ALERT system, since new information sources (e.g. twitter) could be easily integrated. - Complex-Events, Sensor Integration	2012		Scientific Community (higher education, Research)	Europe
131	Collaboration Meeting	Jesús gonzález Barahona (URJC)	Collaboration Working Group on free software - Several virtual meetings have taken place during this period (aprox. one per month). - Free Software	2012		Scientific Community (higher education, Research)	Europe
132	Collaboration Meeting	ATOS	Collaboration Working Group on free software - -	2012		Scientific Community	Europe
133	Video	CIM	Project Promotional video #1: http://alert-project.eu/node/video/6567	2012		Developers, Industrial and Scientific Community	International
134	Video	ATOS	Project Promotional video #2: http://alert-project.eu/node/video/6877	2012		Developers, Industrial and Scientific Community	International

135	Video	FZI	Promotional video interview on Open World Forum Event: http://alert-project.eu/node/video/6883	2012		Developers, Industrial and Scientific Community	International
136	Presentation	ATOS	ALERT Project Presentation http://es.slideshare.net/cpezuela/alert-project-presentation	2011		Developers, Industrial and Scientific Community	International
137	Presentation	ATOS	Shared Online ALERT Presentation at Open World Forum 2012 http://es.slideshare.net/OpenWorldForum/owf12alert-project-workshop	2012		Developers, Industrial and Scientific Community	International
138	Presentation	ATOS	Shared Online ALERT Presentation at fOSSa Event 2012 http://es.slideshare.net/fossaconference/bug-tracking-alert-project-f-ossa2011	2012		Developers, Industrial and Scientific Community	International
139	Web	ATOS	Twitter Account https://twitter.com/alertfp7	2011		Developers, Industrial and Scientific Community	International
140	Web	Alejandra Trujillo (ATOS)	Newsletter #1 coordination and content creation - http://www.alert-project.eu/newsletter/content/alert-newsletter-n%C2%BA1	2011		Developers, Industrial and Scientific Community	International

141	Web	Alejandra Trujillo (ATOS)	Newsletter #2 coordination and content creation - http://www.alert-project.eu/newsletter/content/alert-newsletter-n%C2%BA2	2011		Developers, Industrial and Scientific Community	International
142	Web	Alejandra Trujillo (ATOS)	Newsletter #3 coordination and content creation - http://www.alert-project.eu/newsletter/content/alert-newsletter-n%C2%BA3	2012		Developers, Industrial and Scientific Community	International
143	Web	Oliver Barreto (ATOS)	Newsletter #4 coordination and content creation - http://www.alert-project.eu/newsletter/content/alert-newsletter-n%C2%BA4	2012		Developers, Industrial and Scientific Community	International
144	Web	Oliver Barreto (ATOS)	Newsletter #5 coordination and content creation - http://www.alert-project.eu/newsletter/content/alert-newsletter-n%C2%BA5	2013		Developers, Industrial and Scientific Community	International
145	Web	Alejandra Trujillo & Oliver Barreto (ATOS)	Website revamp: Design, Structure and Content	2011		Developers, Industrial and Scientific Community	International
146	Web	Alejandra Trujillo & Oliver Barreto (ATOS)	Website revamp: Design, Structure and Content Generation	2012		Developers, Industrial and Scientific Community	International

147	Web	Alejandra Trujillo & Oliver Barreto (ATOS)	Website revamp: Design, Structure and Content Generation	2012		Developers, Industrial and Scientific Community	International
148	Software Release	Alejandra Trujillo & Oliver Barreto (ATOS)	Website revamp: Design, Structure and Content Generation	2012		Developers, Industrial and Scientific Community	International
149	Software Release	Alejandra Trujillo & Oliver Barreto (ATOS)	Website preparation for Software Release Design, Structure and Content	2013		Developers, Industrial and Scientific Community	International

Table 3: List of dissemination activities

2.2 Section B (PUBLIC): Exploitation

First off, this section should list any applications for patents, trademarks, registered designs, but since **ALERT consortium has not created any patents during its project life**, the table B1 provided below is empty.

Since ALERT has chosen an Open Source approach since its beginning, we have included all details about its results and expected impact as public content on this document. We have already cited that **ALERT System is an open-source solution that connects with its stakeholders through an open-source public model**, and as already stated before, all components of ALERT System use a permissive nature license: GPL v2, LGPL v3 and BSD. This means that ALERT is offered to the community under an Open Source Public Model through Github.com repository; and that interested industrial organizations within the consortium have the chance to exploit ALERT results selling services based on ALERT's Open Source Software results: installation, configuration, bespoke development of functionality add-ins and sensors, customization, administration, hosting, training, and support. Thus the move to Github.com is very important under the selected exploitation path since it will be the foundation to share the results with the community under a full Open Source approach and fully embracing Social Coding trend enabling customization and evolvement of ALERT components by the software development community.

Github is free to use and represents no operational costs to be taken into account for future sustainability by ALERT partners. Moreover, using Github.com will also allow to take advantage of its capabilities that offer an issue tracking system and a collaboration wiki that could serve as a community hub for interested developers and companies to study and evolve ALERT results, and also to gain developers feedback; and especially by its social coding with ratings, followers (branching and forking), forking and network monitoring capabilities. All this capabilities will enable interested partners to obtain leads to offer ALERT personalized services on demand.

It is worth highlighting that ALERT proposes a solution fully modular for two major reasons:

- Avoiding limitations of each component's license with respect to redistribution, giving the freedom to each component (owner) to keep the type of desired license.
- Diversifying the exploitation paths, allowing users to benefit from ALERT as an overall solution or component by component, depending on their needs.

As consequence, the result of the project is not only a fully integrated platform ready for being downloaded and installed, but also a set of components that can work together following some instructions for installation and configuration, or separately integrated in compatible environments. ALERT can be used in different ways in different situations, addressing different use needs.

2.3.1 Contexts for ALERT

2.3.1.1 ALERT as an standalone system

ALERT can be used as a standalone system, through its own graphical user interface. Since this interface is browser-based, users of the system need only to be provided with a modern web browser to access all the capabilities of the system.

In this case, ALERT has to be deployed in its entirety, linked to the development repositories and support systems of the software projects to which it is to be associated. ALERT will be one of the main entry points to the development activity, allowing developers, project managers and other stakeholders to have different views of the activity of the project. These views will include not only direct access to tickets and code through the search capabilities, but also recommendations, related issues, and much more.

The main value of the system in this case is being a single point of entry, from which all the different aspects of the project are available with an unprecedented level of relationship among them. This is especially useful for organizations that do not already have coordination tools for linking all software development activities. In this case, and in addition to its other capabilities, ALERT can be the main coordination tool for the whole project.

Since ALERT components communicate among themselves using an event bus, adding or removing specific components, and interfacing to specific development repositories or support systems is something that can be done dynamically, according to the needs, and in a very scalable and transparent fashion.

2.3.1.2 ALERT in combination with software forges

An area where ALERT shines specially is in its relationship with already deployed software forges. These days, it is usual that software projects are organized in software forges which provide a coordination system for all the different components supporting the development process. In these cases, ALERT can also be used through its own graphical user interface, as is the case in the standalone scenario. But when developers and users are used to the interface provided by the forge, it is very likely that attracting them to Alert's interface is not an easy task.

In this scenario, the component-based nature of ALERT, with some components providing their own interfaces, designed to simplify the connection with external systems, helps to position it as a simple to use system. For example, all the functionality related to searches is accessible through the REST interface provided by the Search Service. Since this functionality is provided as a JSON-based API, it is easy to write some code that access ALERT from the forge components to enrich the information presented to developers and users.

In this case, the main value of ALERT is its capabilities to enrich the forge with new functionality and information, and how easy that can be done thanks to these interfacing components.

2.3.1.3 ALERT in combination with integrated development environments

The same situation described for the forge can be found in projects (or for developers) which decide to use some integrated development environment (IDE) as the center for all development activities. This happens, for example, when a company decides to standardize on Eclipse and a given set of Eclipse plugins all the development activities. Again, in this case it is not easy to make developers access ALERT through its own interface.

Fortunately, once again ALERT can be used through connecting components. The search service could be used to enrich and enable new queries in the IDE or its plugins, by adding some code to them that access the Search Service API (as was suggested in the forge scenario). But two other approaches could be used to access the system:

- Building a specific IDE plugin that offers a part of the ALERT functionality by interfacing the Search Service via its REST interface. This plugin would be simple and quick to build, and easy to integrate with other plugins providing different kinds of information related to the software development artefacts.
- Building a specific IDE plugin that interfaces directly to ALERT's event bus. In this case, the plugin could access all the functionality of the system. Thanks to the pluggable architecture so usual in IDEs nowadays, this component would be not too difficult to build, and could provide as much functionality as the ALERT user interface itself, but well integrated with the rest of the IDE.

In any of these cases, the value for stakeholders would be in the added functionality that they could obtain, without having to make changes in the usual tool they use to develop the software.

2.3.1.4 ALERT as a component of larger systems (in the foreground)

ALERT can also integrate other systems that could provide information of interest through it. In this scenario, ALERT would itself be considered as a component of a larger system, providing an interface to some of its capabilities, in addition to its own.

An example of this case would be some metrics system that can be integrated (using the Search Service) with information from ALERT services. That way, in addition to showing the regular aspects of the project that ALERT shows, its user interface would also provide linked access to the relevant metrics. For that, the background systems would have to provide simple-to-use interfaces that could be integrated in ALERT.

In this case, the value of ALERT would lie on the intelligence it provides to access the data provided by the background components to which it is linking.

2.3.1.5 ALERT as a component of larger systems (in the background)

More in general, the interfacing components of ALERT can mask it as a component for any kind of larger system, as was proposed in forge scenario. This could allow for example, including an ALERT component in a business intelligence system, making it possible to access most of the information related to software development from it.

The value of ALERT in this scenario would lie on its capabilities to make accessible, in a simple yet very powerful way, software development information to other systems and tools where it could be useful.

2.3.2 Solving Real Problems

ALERT System tries to solve real problem of specific segments by providing means to address the following **Unique Value Propositions (UVPs)** or dimensions to position ALERT results on the market:

- **Development Process Improvement & Collaboration:** integration of system generated project knowledge base and notification systems to coding and collaboration tools with automated extraction of relevant information from several web 2.0 and coding tools data sources.
- **Enhance Development Tools:** Tools that are either used as an information data source for ALERT system or as UI to notify and interact with the user.
- **Embrace and Empower Social Coding:** Automated creation of a life-time Developer Profile with real skills & contribution to projects (Developers can automatically build real profiles to show off their coding skills, expertise and contributions & Companies can internally manage and find external talent, skills for recruiting purposes).
- **Help Talent be Recruited:** very related with social coding but in this case, not for showing off, but for letting companies know that they can access developer profiles with real skills & contribution to projects.

Accordingly, different messages have been crafted and made available through the website landing pages and also by social media channels.

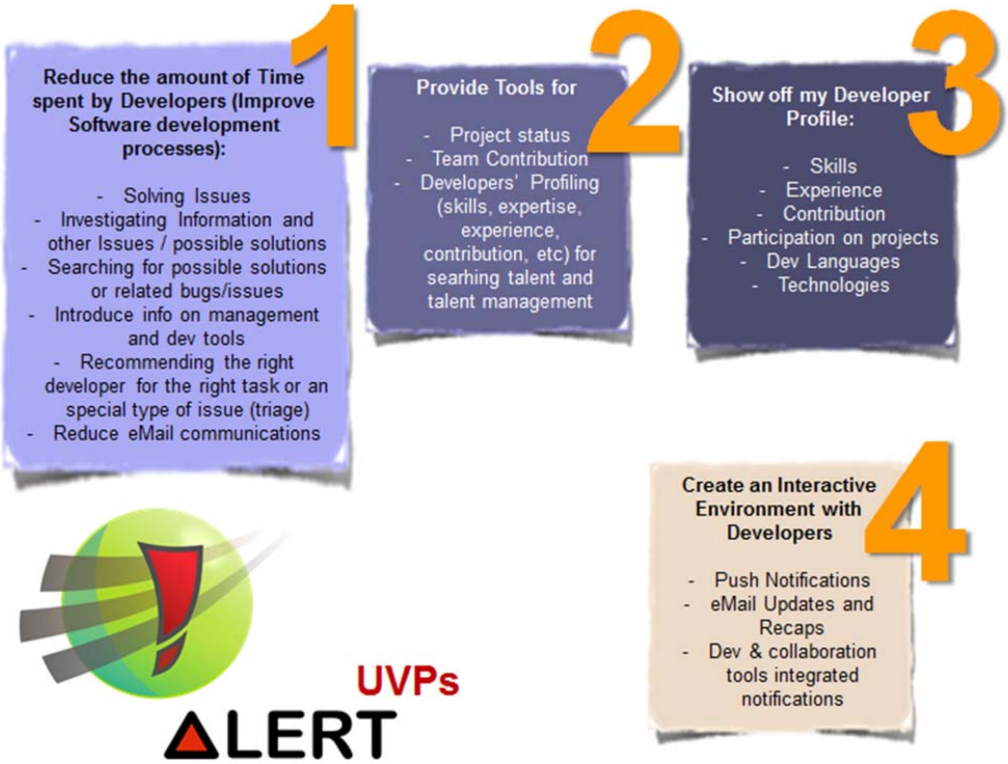

Figure 14: ALERT UVPs used for Business Scenarios Modeling

We must also explicitly mention here the fact that ALERT consortium foresees great potential in the future for taking actions to evolve related technology based on the results of ALERT project or to create a company that exploits ALERT results commercially in the following ways:

- Building software that improves automation of gathering and evaluating developer's profiles and offering services for recruiting purposes, for talent management or for salary calculations. This technology can even be further improved by extending its scope to not only developers by to other types of job descriptions that interact with multiple tools and systems, especially on social media software for corporations.
- Studying how to infer information and extract knowledge and trends from user behaviour, interactions and the created knowledge base that could be notified back to different multi-disciplinary agents. Therefore, creating a meta-library that can be accessed by multiple companies that for instance want to have access to information about bugs or issues that are related to their hardware or software, and that they can use to infer trends or patterns that affect their business. On this scenario, the role of a broker should be created to ensure that the information extracted from different ALERT repositories is only used to offer those services and in a secured an anonymous manner.

2.3.3 Data Tables

We provide next Table B1 with data regarding Patents, Trademarks and Registered Designs and Table B2 with an overview of exploitable results.

TABLE B1: LIST OF APPLICATIONS FOR PATENTS, TRADEMARKS, REGISTERED DESIGNS, ETC. Type of IP Rights: Patents, Trademarks, Registered designs, Utility models, Others.					
Type of IP Rights (Patents, Trademarks, Registered designs, Utility models, Others)	Confidential (YES/NO)	Foreseen embargo date (dd/mm/yyyy)	Application reference(s) (e.g. EP123456)	Subject or title of application	Applicant(s) (as on the application)
N/A					

Table 4: List of application for patents, trademarks, registered designs

TABLE B2: OVERVIEW TABLE WITH EXPLOITABLE FOREGROUND

Type of Exploitable Foreground: General advancement of knowledge, Commercial exploitation of R&D results, Exploitation of R&D results via standards, exploitation of results through EU policies, exploitation of results through (social) innovation.

Sectors of Application: according to the following nomenclature at http://ec.europa.eu/competition/mergers/cases/index/nace_all.html

NO.	Type of Exploitable Foreground	Description of exploitable foreground	Confidential (YES/NO)	Foreseen embargo date (dd/mm/yyyy)	Exploitable product(s) or measure(s)	Sector(s) of application	Timetable, commercial or any other use	Patents or other IPR exploitation (licences)	Owner & Other Beneficiary(s) involved
1	Component	Knowledge extractor for structured information (KESI)Extraction of knowledge from structured data sources (SCM, BTS)	NO		Easier interpretation of data and context extracted from structured sources.. Standalone or as a service	Software		GPL V2+	URJC
2	Component	Knowledge extractor for unstructured information (KEUI)KEUI is responsible for semantically enriching all textual posts (mails, bug reports, forum posts, etc.) from all monitored data sources. It also stores the posts in order to support information retrieval.	NO		KEUI plays a very important part since it process unstructured text which is one of the main sources of information in ALERT. . as a service or packaged	Software		GPL v3	JSI

3	Component	Metadata ServiceMetadata service provides support for creation, retrieval, reasoning about and storage of ontology-based metadata. Additionally, it triggers semantically-enriched event-based communication.	NO		This service is shared by all other components/services as it provides an abstraction from the ontology triplets to the level of ALERT entities such as bugs, users, methods, etc.. Standalone or as a service	Software		AGPL v3	URJC
4	Component	Profiler (STARDOM)STARDOM is responsible for creating and maintaining a developer's profile. More specifically, it focuses on enriching the profile with information about the expertise of the developer. The expertise of the developer is modelled and stored in a competency model.	NO		The fact that STARDOM uses a wide variety of metrics to calculate the Competency Index results in a more accurate profile than profiles built by existing techniques. This component can be easily extended to incorporate other different sources of data to be later considered for calculating the profile index based on several categories which can be invaluable to open exploitation opportunities.. as a service; deployed in a platform	Software		GPL v2+	ICCS

5	Component	CEP EngineETALIS is event processing framework which includes an expressive language for complex patterns, and an efficient execution model. The framework is logic-based, and feature deductive capabilities exploitable in event processing, as well as, in (re-)actions and workflows triggering.	NO		Provision of the right information to right people/components in right situations. This is a key issue in commercial situation when notifying users without being considered obtrusive but at the same time being interactive enough.. It can be exploited either as a service or a standalone component	Software		LGPL v3	FZI
6	Component	Interaction Pattern EditorThe interaction pattern editor enables the developers to describe situations relevant for notification. It is the implementation of a methodology for the management of complex interaction patterns	NO		Supporting the domain expert in creating new patterns, increasing the reusability of existing patterns, refining patterns and enabling a semi-automatic pattern evolution. as a service or packaged	Software		LGPL v3	FZI
7	Component	Action serviceThis service helps users to subscribe to the different notifications which are provided by ALERT system	NO		Execution of the actions related to the detected situations. as a service; deployed in a platform	Software		LGPL v3	FZI

8	Component	ALERT System REST APIThe ALERT's REST API offers support for simple queries and search capabilities exploring interconnections among knowledge elements retrieved from ITS, SCM and other data sources. For example, ALERT users are able to explore possible duplicates for a given issue report, other issues sharing certain characteristics (same module, similar topic) and some connections between issues and related source code elements.	NO		This is the first attempt to merge information from different sources of knowledge into coherent indicators to explore links and connections among these knowledge elements.. N/A	Software		N/A	All
9	Component	Recommendation serviceThe recommendation service is used in order to suggest developers that are considered the most knowledgeable to resolve an issue. It also suggests issues to developers according to their areas of expertise and competency indexes	NO		Automatic bug triage and avoidance of cost and time loss. Essential for managers to exploit ALERT results in real scenarios of development teams (large or geographically dispersed). as a service; deployed in a platform	Software		GPL v2+	ICCS

10	Main User Interface to interact with ALERT system	Visualization serviceVisualization of the results of the use-cases and provides a visual UI tool to interact with the system.	NO		The UI tool in conjunction with the Visualization service will improve the usability of the ALERT system by allowing the users to be more efficient in their work.. as a service or packaged	Software		GPL v3	JSI
11	Sensor	SCM sensorSynchronous extraction of information from SCM	NO		Real-time tracking of SCM updates. Standalone or as a service	Software		GPL V2+	URJC
12	Sensor	BTS sensorSynchronous extraction of information from BTS	NO		Real-time tracking of BTS updates. Standalone or as a service	Software		GPL V2+	URJC
13	Sensor	Wiki SensorSynchronous extraction of information from Wiki	NO		Real-time tracking of wiki updates. Standalone or as a service	Software		Apache License, Version 2.0	FZI
14	Sensor	Mailing List SensorSynchronous extraction of information from Mailing List	NO		Real-time tracking of Mailing List updates. Runs as a stand alone service	Software		GPL v3	ICCS

15	Sensor	Forum SensorSynchronous extraction of information from Forum	NO		Real-time tracking of Forum updates. Runs as a stand alone service	Software		MS-PL	JSI
16	Sensor	OcelotOnline Semantic Concept Extractor based on Linked Open Data	NO		Real-time tracking of semantic data. as a service or packaged	Software		GPL V2+	ATOS

Table 5: Overview Table with Exploitable Foreground

3. Report on societal implications

We include next ALERT figures for the expected questionnaire to assist the Commission to obtain statistics and indicators on societal and socio-economic issues addressed by projects.

A General Information *(completed automatically when **Grant Agreement number** is entered.*

Grant Agreement Number:	258098
Title of Project:	Active support and reaL-time coordination based on Event pRocessing in open source software development
Name and Title of Coordinator:	Dr. Ljiljana Stojanovic

B Ethics

1. Did your project undergo an Ethics Review (and/or Screening)? • If Yes: have you described the progress of compliance with the relevant Ethics Review/Screening Requirements in the frame of the periodic/final project reports? Special Reminder: the progress of compliance with the Ethics Review/Screening Requirements should be described in the Period/Final Project Reports under the Section 3.2.2 *'Work Progress and Achievements'*	***Yes No***
2. Please indicate whether your project involved any of the following issues (tick box) :	***No***
RESEARCH ON HUMANS	
• Did the project involve children?	
• Did the project involve patients?	
• Did the project involve persons not able to give consent?	
• Did the project involve adult healthy volunteers?	
• Did the project involve Human genetic material?	
• Did the project involve Human biological samples?	
• Did the project involve Human data collection?	
RESEARCH ON HUMAN EMBRYO/FOETUS	
• Did the project involve Human Embryos?	
• Did the project involve Human Foetal Tissue / Cells?	
• Did the project involve Human Embryonic Stem Cells (hESCs)?	
• Did the project on human Embryonic Stem Cells involve cells in culture?	
• Did the project on human Embryonic Stem Cells involve the derivation of cells from Embryos?	
PRIVACY	
• Did the project involve processing of genetic information or personal data (eg. health, sexual lifestyle, ethnicity, political opinion, religious or philosophical conviction)?	
• Did the project involve tracking the location or observation of people?	
RESEARCH ON ANIMALS	

• Did the project involve research on animals?	
• Were those animals transgenic small laboratory animals?	
• Were those animals transgenic farm animals?	
• Were those animals cloned farm animals?	
• Were those animals non-human primates?	
RESEARCH INVOLVING DEVELOPING COUNTRIES	
• Did the project involve the use of local resources (genetic, animal, plant etc)?	
• Was the project of benefit to local community (capacity building, access to healthcare, education etc)?	
DUAL USE	
• Research having direct military use	Yes No
• Research having the potential for terrorist abuse	

C Workforce Statistics

3. Workforce statistics for the project: Please indicate in the table below the number of people who worked on the project (on a headcount basis).

Type of Position	Number of Women	Number of Men
Scientific Coordinator	1	4
Work package leaders		7
Experienced researchers (i.e. PhD holders)	10	16
PhD Students	2	14
Other	6	20

4. How many additional researchers (in companies and universities) were recruited specifically for this project?	
Of which, indicate the number of men:	7
Of which, indicate the number of women:	5

D Gender Aspects

5. Did you carry out specific Gender Equality Actions under the project?

○ Yes
⦿ No

6. Which of the following actions did you carry out and how effective were they?

		Not at all effective				Very effective
❑	Design and implement an equal opportunity policy	○	○	○	○	○
❑	Set targets to achieve a gender balance in the workforce	○	○	○	○	○
❑	Organise conferences and workshops on gender	○	○	○	○	○
❑	Actions to improve work-life balance	○	○	○	○	○

○ Other:

7. Was there a gender dimension associated with the research content – i.e. wherever people were the focus of the research as, for example, consumers, users, patients or in trials, was the issue of gender considered and addressed?

○ Yes- please specify

⦿ No

E Synergies with Science Education

8. Did your project involve working with students and/or school pupils (e.g. open days, participation in science festivals and events, prizes/competitions or joint projects)?

○ Yes- please specify

⦿ No

9. Did the project generate any science education material (e.g. kits, websites, explanatory booklets, DVDs)?

⦿ Yes- please specify: We created a website providing the ALERT demo and are working on a webpage providing information about how to use the system

○ No

F Interdisciplinarity

10. Which disciplines (see list below) are involved in your project?

⦿ Main discipline: 1.1

○ Associated discipline: ○ Associated discipline:

G Engaging with Civil society and policy makers

11a Did your project engage with societal actors beyond the research community? *(if 'No', go to Question 14)*

○ Yes
⦿ No

11b If yes, did you engage with citizens (citizens' panels / juries) or organised civil society (NGOs, patients' groups etc.)?

- ○ No
- ○ Yes- in determining what research should be performed
- ○ Yes - in implementing the research
- ○ Yes, in communicating /disseminating / using the results of the project

11c In doing so, did your project involve actors whose role is mainly to organise the dialogue with citizens and organised civil society (e.g. professional mediator; communication company, science museums)?	○ ○	Yes No

12. Did you engage with government / public bodies or policy makers (including international organisations)

- ◉ No
- ○ Yes- in framing the research agenda
- ○ Yes - in implementing the research agenda
- ○ Yes, in communicating /disseminating / using the results of the project

13a Will the project generate outputs (expertise or scientific advice) which could be used by policy makers?

- ○ Yes – as a **primary** objective (please indicate areas below- multiple answers possible)
- ○ Yes – as a **secondary** objective (please indicate areas below - multiple answer possible)
- ◉ No

13b If Yes, in which fields?

Agriculture Audiovisual and Media Budget Competition Consumers Culture Customs Development Economic and Monetary Affairs Education, Training, Youth Employment and Social Affairs		Energy Enlargement Enterprise Environment External Relations External Trade Fisheries and Maritime Affairs Food Safety Foreign and Security Policy Fraud Humanitarian aid		Human rights Information Society Institutional affairs Internal Market Justice, freedom and security Public Health Regional Policy Research and Innovation Space Taxation Transport	

13c If Yes, at which level?

- ○ Local / regional levels
- ○ National level
- ○ European level
- ○ International level

H Use and dissemination

14. How many Articles were published/accepted for publication in peer-reviewed journals?	**21**
To how many of these is open access provided?	
How many of these are published in open access journals?	**21**
How many of these are published in open repositories?	**21**
To how many of these is open access not provided?	**0**
Please check all applicable reasons for not providing open access:	
❑ publisher's licensing agreement would not permit publishing in a repository ❑ no suitable repository available ❑ no suitable open access journal available ❑ no funds available to publish in an open access journal ❑ lack of time and resources ❑ lack of information on open access ❑ other: ……………	
15. How many new patent applications ('priority filings') have been made? *("Technologically unique": multiple applications for the same invention in different jurisdictions should be counted as just one application of grant).*	**0**

16. Indicate how many of the following Intellectual Property Rights were applied for (give number in each box).	Trademark	**0**
	Registered design	**0**
	Other	**0**

17. How many spin-off companies were created / are planned as a direct result of the project?	**1**
Indicate the approximate number of additional jobs in these companies:	**3**

18. Please indicate whether your project has a potential impact on employment, in comparison with the situation before your project:

❑ Increase in employment, or	⦿ In small & medium-sized enterprises
⦿ Safeguard employment, or	⦿ In large companies
❑ Decrease in employment,	❑ None of the above / not relevant to the project
❑ Difficult to estimate / not possible to quantify	

19. For your project partnership please estimate the employment effect resulting directly from your participation in Full Time Equivalent (*FTE = one person working fulltime for a year*) jobs:	*Indicate figure:*
Difficult to estimate / not possible to quantify	⦿

I Media and Communication to the general public

20. As part of the project, were any of the beneficiaries professionals in communication or media relations?

○ Yes ⦿ No

21. As part of the project, have any beneficiaries received professional media / communication training / advice to improve communication with the general public?

⦿ Yes ○ No

22 Which of the following have been used to communicate information about your project to the general public, or have resulted from your project?

⦿	Press Release	⦿	Coverage in specialist press
⦿	Media briefing	⦿	Coverage in general (non-specialist) press
⦿	TV coverage / report	❑	Coverage in national press
❑	Radio coverage / report	❑	Coverage in international press
⦿	Brochures /posters / flyers	⦿	Website for the general public / internet
⦿	DVD /Film /Multimedia	⦿	Event targeting general public (festival, conference, exhibition, science café)

23 In which languages are the information products for the general public produced?

❑	Language of the coordinator	⦿	English
⦿	Other language(s)		

Question F-10: Classification of Scientific Disciplines according to the Frascati Manual 2002 (Proposed Standard Practice for Surveys on Research and Experimental Development, OECD 2002):

FIELDS OF SCIENCE AND TECHNOLOGY

1. NATURAL SCIENCES

1.1 Mathematics and computer sciences [mathematics and other allied fields: computer sciences and other allied subjects (software development only; hardware development should be classified in the engineering fields)]

1.2 Physical sciences (astronomy and space sciences, physics and other allied subjects)

1.3 Chemical sciences (chemistry, other allied subjects)

1.4 Earth and related environmental sciences (geology, geophysics, mineralogy, physical geography and other geosciences, meteorology and other atmospheric sciences including climatic research, oceanography, vulcanology, palaeoecology, other allied sciences)

1.5 Biological sciences (biology, botany, bacteriology, microbiology, zoology, entomology, genetics, biochemistry, biophysics, other allied sciences, excluding clinical and veterinary sciences)

2 ENGINEERING AND TECHNOLOGY

2.1 Civil engineering (architecture engineering, building science and engineering, construction engineering, municipal and structural engineering and other allied subjects)

2.2 Electrical engineering, electronics [electrical engineering, electronics, communication engineering and systems, computer engineering (hardware only) and other allied subjects]

2.3. Other engineering sciences (such as chemical, aeronautical and space, mechanical, metallurgical and materials engineering, and their specialised subdivisions; forest products; applied sciences such as geodesy, industrial chemistry, etc.; the science and technology of food production; specialised technologies of interdisciplinary fields, e.g. systems analysis, metallurgy, mining, textile technology and other applied subjects)

3. MEDICAL SCIENCES

3.1 Basic medicine (anatomy, cytology, physiology, genetics, pharmacy, pharmacology, toxicology, immunology and immunohaematology, clinical chemistry, clinical microbiology, pathology)

3.2 Clinical medicine (anaesthesiology, paediatrics, obstetrics and gynaecology, internal medicine, surgery, dentistry, neurology, psychiatry, radiology, therapeutics, otorhinolaryngology, ophthalmology)

3.3 Health sciences (public health services, social medicine, hygiene, nursing, epidemiology)

4. AGRICULTURAL SCIENCES

4.1 Agriculture, forestry, fisheries and allied sciences (agronomy, animal husbandry, fisheries, forestry, horticulture, other allied subjects)

4.2 Veterinary medicine

5. SOCIAL SCIENCES

5.1 Psychology

5.2 Economics

5.3 Educational sciences (education and training and other allied subjects)

5.4 Other social sciences [anthropology (social and cultural) and ethnology, demography, geography (human, economic and social), town and country planning, management, law, linguistics, political sciences, sociology, organisation and methods, miscellaneous social sciences and interdisciplinary , methodological and historical S1T activities relating to subjects in this group. Physical anthropology, physical geography and psychophysiology should normally be classified with the natural sciences].

6. HUMANITIES

6.1 History (history, prehistory and history, together with auxiliary historical disciplines such as archaeology, numismatics, palaeography, genealogy, etc.)

6.2 Languages and literature (ancient and modern)

6.3 Other humanities [philosophy (including the history of science and technology) arts, history of art, art criticism, painting, sculpture, musicology, dramatic art excluding artistic "research" of any kind, religion, theology, other fields and subjects pertaining to the humanities, methodological, historical and other S1T activities relating to the subjects in this group]

References

"Wordnet," http://wordnet.princeton.edu/. [Online]. Available: http://wordnet.princeton.edu/.

"Webopedia," 2011. [Online]. Available: http://www.webopedia.com/.

"whatis.techtarget.com," 2011. [Online]. Available: http://whatis.techtarget.com/.

"StackOverflow," 2011. [Online]. Available: http://stackoverflow.com/.

[1] S. Gauch, M. Speretta, A. Chandramouli, and A. Micarelli, "User profiles for personalized information access," *The adaptive web*, pp. 54–89, 2007.

[2] F. Carmagnola and F. Cena, "User identification for cross-system personalisation," *Information Sciences*, vol. 179, no. 1–2, pp. 16–32, 2009.

[3] G. Jeong, S. Kim, and T. Zimmermann, "Improving bug triage with bug tossing graphs," in *Proceedings of the the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*, 2009, pp. 111–120.

[4] F. Abel, N. Henze, E. Herder, and D. Krause, "Interweaving Public User Profiles on the Web," in *International Conference on User Modeling, Adaptation and Personalization (UMAP), Hawaii, USA. Springer*, 2010.

[5] R. C. Lange and S. Mancoridis, "Using code metric histograms and genetic algorithms to perform author identification for software forensics," in *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, 2007, p. 2089.

[6] R. Iváncsy and S. Juhász, "Analysis of Web User Identification Methods," *International Journal of Computer Science*, vol. 2, no. 3, pp. 212–219, 2007.

[7] G. Robles and J. M. Gonzalez-Barahona, "Developer identification methods for integrated data from various sources," in *Proceedings of the 2005 international workshop on Mining software repositories*, 2005, pp. 1–5.

[8] O. Alonso, P. T. Devanbu, and M. Gertz, "Expertise identification and visualization from CVS," in *Proceedings of the 2008 international working conference on Mining software repositories*, 2008, pp. 125–128.

[9] A. Vetro, M. Torchiano, and M. Morisio, "Assessing the precision of FindBugs by mining Java projects developed at a university," in *Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on*, pp. 110–113.

[10] T. Fritz, G. C. Murphy, and E. Hill, "Does a programmer's activity indicate knowledge of code?," in *Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*, 2007, pp. 341–350.

[11] A. Mockus and J. D. Herbsleb, "Expertise browser: a quantitative approach to identifying expertise," in *Proceedings of the 24th International Conference on Software Engineering*, 2002, pp. 503–512.

[12] H. Kagdi, M. Hammad, and J. I. Maletic, "Who can help me with this source code change?," in *Software Maintenance, 2008. ICSM 2008. IEEE International Conference on*, 2008, pp. 157–166.

[13] J. Geldenhuys, "Finding the Core Developers," in *2010 36th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA)*, 2010, pp. 447–450.

[14] J. Anvik, "Automating bug report assignment," in *Proceedings of the 28th international conference on Software engineering*, New York, NY, USA, 2006, pp. 937–940.

[15] D. Schuler and T. Zimmermann, "Mining usage expertise from version archives," in *Proceedings of the 2008 international working conference on Mining software repositories*, 2008, pp. 121–124.

[16] A. Moraes, E. Silva, C. da Trindade, Y. Barbosa, and S. Meira, "Recommending experts using communication history," in *Proceedings of the 2nd International Workshop on Recommendation Systems for Software Engineering*, New York, NY, USA, 2010, pp. 41–45.
[17] D. Matter, A. Kuhn, and O. Nierstrasz, "Assigning bug reports using a vocabulary-based expertise model of developers," in *Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on*, 2009, pp. 131–140.
[18] H. Kagdi and D. Poshyvanyk, "Who can help me with this change request?," in *IEEE 17th International Conference on Program Comprehension, 2009. ICPC '09*, 2009, pp. 273 –277.
[19] X. Xie, W. Zhang, Y. Yang, and Q. Wang, "DRETOM: developer recommendation based on topic models for bug resolution," in *Proceedings of the 8th International Conference on Predictive Models in Software Engineering*, New York, NY, USA, 2012, pp. 19–28.
[20] M. C. De Marneffe, B. MacCartney, and C. D. Manning, "Generating typed dependency parses from phrase structure parses," in Proceedings of LREC, 2006, vol. 6, pp. 449–454.
[21] K. Toutanova and C. D. Manning, "Enriching the knowledge sources used in a maximum entropy part-of-speech tagger," in Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics-Volume 13, 2000, pp. 63–70.
[22] M. F. Porter and others, An algorithm for suffix stripping. Program, 1980.
[23] V. V. Raghavan and S. Wong, "A critical analysis of vector space model for information retrieval," Journal of the American Society for Information Science, vol. 37, no. 5, pp. 279–287, 1986.
[24] P. Runeson, M. Alexandersson, and O. Nyholm, "Detection of duplicate defect reports using natural language processing," in Proceedings of the 29th international conference on Software Engineering, 2007, pp. 499–510.
[25] D. C. Luckham. The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA. 2001.
[26] O. Etzion und P. Niblett. Event Processing in Action. Manning Publications Co, ISBN-10: 1935182218, 2010.
[27] S. Rozsnyai, J. Schiefer und A. Schatten. Concepts and models for typing events for event-based systems. In DEBS '07: Proceedings of the 2007 inaugural international conference on Distributed event-based systems, New York, NY, USA, 2007. ACM, S. 62–70
[28] D. Snowden. Organic Knowledge Management: Part I The ASHEN Model: an enabler of action. Knowledge Management 3(7), 2004
[29] S. Wasserkrug, A. Gal, O. Etzion und Y. Turchin. Complex event processing over uncertain data. In Proceedings of the second inter-national conference on Distributed event-based systems, DEBS '08, New York, NY, USA, 2008. ACM, S. 253–264.
[30] David Luckham. What is the difference between ESP and CEP? 2007, http://www.complexevents.com/2006/08/01/what%E2%80%99s-the-differencebetween-esp-and-cep/
[31] Turchin, Yulia and Gal, Avigdor and Wasserkrug, Segev. Tuning complex event processing rules using the prediction-correction paradigm. Proceedings of the Third ACM International Conference on Distributed Event-Based Systems, 2009