Consultation on Future Research Priorities on Software & Services

Welcome to the On-line Consultation process of the European Commission, DG Information Society and Media, Service and Software Architectures and Infrastructures Unit (INFSO D3). This is a consultation on the 7th Framework programme of Research and Development in Information and Communication Technologies.

Through this consultation process you are able to provide your input to the definition of the Work Programme for 2011-2013 in the field of software and services. This Work Programme will be the basis for the next calls of proposal in this area.

The forum-based consultation tool allows you to provide your comments, contributions and suggestions in a simple format. Inserted contributions can be publicly displayed on-line in one of two ways – either as an anonymous contribution or including your name and organisation. Or you can choose to include your contribution in the data provided to the EC but not be publicly visible on-line.

The forum will be open for contributions until November 2nd, 2009. After that, the European Commission will consult with other parties, including the member states. The final Work Programme is expected to be announced in the course of 2010.

The European Commission services are pleased to acknowledge the facilities offered by ECSS for its consultation. ECSS will not filter or process the data that is submitted to this consultation, but will deliver it unchanged to the Commission.

Contributions

- **2009-11-02 19:41:38**
  Enterprise Architectures and Value Networks In the future business activities are increasingly done by networks of actors having business, service and infrastructure relationships. Business processes cross often company borders and software as a service and cloud computing paradigms get more widely used. This development calls for new way of architecture management and governance. Research is needed to develop future oriented architectures and process management etc for complex web of business activities realised by ecosystems of service providers.

- **Achterberg Joeri (Twynstra Gudde), 2009-11-02 15:44:57**
  There is a chasm between the potential and the benefits of open source software and its current adoption. In recent years much research has gone into open source software itself: in its development model, the software, the structure of open source communities and the business model. This research has demonstrated time and again that many open source software programs are qualitatively good and the development model is resilient. Also, there are numerous sound economic and political reasons for governments to stimulate the adoption of open source. Nevertheless, the adoption of open source software is slow. We would like to see research done to close the chasm. We believe such research should focus on the reasons organisations have to not adopt open source software. The research will have to take many disciplines into account:
technical, economical, organizational, judicial, etc. The innovation part of this research should, however, not be the analysis of current affairs. It should move beyond and develop innovative solutions to remove these barriers. Solutions can be technical, but need not be. It is more likely that other barriers are stronger and more difficult overcome. Ideally, research efforts on this topic should combine technical expertise with judicial, economical, behavioural and other relevant disciplines. What is the relevance for the EU? - Policy: this initiative matches goals / is meant to achieve goals. - Economy: EU position in ICT can be made stronger (against upcoming ICT regions like India and UAE). - Enhancement of innovation position in the world - Economic stimulation through knowledge

2009-11-02 14:00:53
If we go back to the influential 1999 USA PITAC (President's Information Technology Advisory Committee) report, we notice that one of the key findings was that investment in IT R&D is too focused on near-term problems. I think that European Research on Software and Services suffers today from exactly the same problem. There is an abundance of software infrastructures, which are monolithic, fragile, and require lots of effort to interoperate with different technologies. Their maintenance has a high cost and, way too often, lots of money is invested to add patches. I think software as a whole needs radically new ideas and a significant investment in fundamental, long-term research to guarantee future European leadership. The arguments made 10 years ago by the PITAC committee apply equally well to Europe today: "make fundamental software research an absolute priority"!

2009-11-01 16:04:07
Most researchers nowadays agree that current and future software systems and services should be considered as loosely coupled interacting heterogeneous agents, which must collaborate on a trustful environment. Like in society, the interaction of such agents in SOA needs to be governed by normative contracts. Such contracts are to be understood in a broad sense, including what is the expected normal behavior, and also what is to be done in exceptional cases and what are the penalties in case of misbehavior. Such contracts must be concerned with functional, as well as non-functional/quantitative properties, including for instance real-time aspects (performance, availability, etc). These contracts must act at different levels: from how we program services, e.g. at the object and component level, to the business and decision making level. They must also regulate negotiation when a service is requested, composition of services, and of course monitoring during the real interaction between service provider and consumer. As systems evolve during time, it is naturally to think that services also evolve, and with them the corresponding service contracts. For example, a service offering to download music to a mobile phone may evolve through the form of new promotions. These promotions may then overwrite previous rights by extending them, or simply giving the option to the user to choose between a previous offer and a (hopely) more attractive new one. If the user agrees with accepting the promotion, then the old contract is then modified by adding and/or deleting clauses. At the end of the day the user wants to be sure that she got what she paid for, and this should be guaranteed by the corresponding current contract. The challenge is then to be able to handle evolution of services together with evolution of contracts. This calls for a a contract-based understanding and development philosophy of communicating information systems. Moreover, there is a need to address foundational and practical questions concerning the notion of contracts without being decoupled from the notion of services. Also, a contract-aware programming scheme
would be desirable, in order to program services already having in mind the dynamic aspect. We also need to provide development tools, and to develop specification and verification techniques that support efficient production of evolvable systems and contracts.

- **van ‘t Veld Steven (A/I/M bv), 2009-11-01 15:22:31**

   At present and in the future the real issue in the information and IT sector will be information, not IT. Reasons are simple: ... Organisations only invest in (administrative) IT because they need information. ... Information is not an IT issue. Even further: IT professionals treat information from a technology perspective. For that reason it is a big problem for them, while if it is treated as a business issue it usually becomes quite understandable, a very powerful instrument to have as a corporate resource. Although further development of IT is very necessary and relevant, the main issue is to determine what data is information in organisations. There is only marginal research and development on information, and this will limit the strength of our organisations in the near future, and therefore will affect our market effectiveness more and more.

- **Cacciari Claudio (CINECA), 2009-10-27 12:07:02**

   In my opinion, an interesting research topic is related to the message flow analysis and presentation. There is an increasing attention about people message flow management (for example see Google Wave, https://wave.google.com/wave or Mozilla Weave and Raindrop, http://labs.mozilla.com/projects/) and similar approaches could be applied to service messages also. Even if the Cloud hosting environments offer API to monitor the status of the services, the message flow monitoring could contribute to give to the service administrators a single point of view on all their services, scattered among heterogeneous systems. The possibility to control the service message flow would allow a user to consider his/her service requests on the same level of mails, with search and filter capabilities. Moreover the development of a data flow analysis in service environments could open the way to other interesting experiments such as the use of infographics (http://en.wikipedia.org/wiki/Information_graphics) concepts in those contexts which could improve the user interaction; or the dynamic creation of services based on the requests and the expected responses.

- **Kuin Klaas (oce-technologies b.v.), 2009-10-26 12:05:16**

   In our opinion services development is strongly related to the customer environment; so we think that the interaction with customers and how to describe services should be understandable by them. we intend to do services development in interaction with customers (and most of the time at their premises) to be able to measure results and successes. So also the customer activities in this field should be fundable. what we have in mind with respect to content of our plans at least the following seem relevant: 1. the (change in ) user interest in paper based vs. electronic information gathering, handling and management. also the balance between the two will be relevant. 2. the way to describe these services both in customer as in ICT language 3. understanding what change in revenue model are possible/necessary 4. learning to realise that some functionality should be delivered by the own organisation and some by partners (which ones and why?)

- **Migliardi Mauro (), 2009-10-19 13:59:37**

   Why Software Engineering is not Engineering Yet? Recently in the field of software engineering there is a rather diffused perception that traditional engineering thinking is
not productive in the software field. In an oversimplified version, it is possible to state
that Software Engineering may not be engineering after all. At least, not yet. In fact,
the evolution from the waterfall model toward iterative, agile methodologies has
stressed that the typical engineering view of planning everything from the beginning
to pave a predictable, clear path to success cannot survive the high level of uncertainty
that plagues the field of software production. It is a common experience to have
components that, although perfectly compatible from the syntactical interface point of
view, when integrated reveal unexpected and pathologically disruptive side effects.
Some authors argue that this is a reason to move from engineering approaches toward
economy like approaches based on a steer-as-you-go behavior, and this has actually
had the benign effect of forcing software projects to integrate system parts as soon as
possible to expose inconsistencies before a major scrap and rewrite effort is needed.
Nonetheless, our position is quite the opposite: the solution is not to abandon the
engineering approach because of high level of uncertainty in software component
interaction, on the contrary, the solution lies in a deeper architectural knowledge of the
system and the software platforms involved to greatly reduce the above mentioned
uncertainty. A typical artifact of the engineering mindset is the use of black box
components. To build a bridge, an engineer does not need to know all the nitty-gritty
details of how a girder is built, he relies on the component data-sheet. This is a tried
and true approach, nonetheless it shows an unpleasantly high failure rate when applied
to software components. We argue that this is due to the fact that the equivalent of a
girder data-sheet is a set of UML diagrams that describes only the formal aspects of
the component with no relation to its architectural behavior (the “implementation is
not an issue” tenet) and even those are strictly valid only as long as the component is
used exactly in the expected way. Thus our position is that to achieve a true “Software
Engineering” we need a new generation of component definition tools and a
systematic study of the effects of run-time platforms on the behavior of software
components.

• **Barioglio Maurizio** (Value Team spa), 2009-10-08 12:21:59
In my opinion the ECSS white paper currently does not cover a fundamental aspect:
the development of agile and dynamic composite applications. I try to briefly explain
my statement. In a SOA ecosystem we can distinguish two main components: 1) a
service exposure layer, 2) composite applications that execute business tasks using the
services offered by exposure layer. The ECSS Whitepaper covers very well, in my
opinion, the service exposure layer. However, if also the other component (composite
applications) is not agile and dynamic the whole ecosystem won’t be. Here is an
example: very often a composite application developed for a company (a Telco
operator, an Insurance Company, a Bank, a Manufacturer) or for a public authority is
strictly dependent on the characteristics of the specific products/services they offer to
customers/citizens. The product/service business logic is often “hardcoded” in the
programming code. Simple products/services characteristics can be described with
some kind of parameterization but the complex ones require coding, today. This brings
to long time-to-market, costs, inflexibility. BPM languages like BPMN and BPEL
fulfil the need to describe business logic in terms of process activities to execute, but
do not cover many other aspects (i.e. the intrinsic characteristics of products/services
like interdependencies among optional product components, pricing, user presentation,
billing, provisioning, shop floor control, ...). I tried to explain this concept in my blog:
http://mauriziobarioglio.wordpress.com/2009/05/19/bpm-is-a-too-tight-dress-on-soa/.
In my opinion researchers should investigate how to model complex business logic in
order to obtain immediate IT solutions for the business specifications without software
coding. I have in mind the CAD-CAM paradigm: as soon as you model an object you can produce it. Moreover business models should be non-procedural (explain “what” is required and not “how” to obtain it) in order to build dynamic adaptation to exceptions and therefore self-healing solutions. BPMN and BPEL are procedural, inflexible languages. As far as I know MDA covers only partially these requirements and focuses on code generation. In my opinion models should be directly executed and call existing, reusable SOA services. Moreover different aspects of business require different but consistent representations. A sort of “UML for business modelling” is in my dreams.

- **2009-09-22 16:40:16**
  The cloud paradigm was mainly conceived and implemented by industrial players. The IT scientific community had a secondary role in this area. The delay of the EU research community in this field should be overcome with a significant effort of EC and the scientific community cooperation by implementing R&D activities with a long term horizon that aim to provide new visions and solutions in the area of cloud computing and services. New models, architectures and services for inter-clouds for e-science, pervasive architectures embodying clouds and ubiquitous devices, and knowledge-oriented clouds must be designed to support disruptive solutions for future user communities and innovative ecosystems. Service oriented solutions able to address integrated uses of clouds and pervasive systems need to be addressed as contributions to Future Internet.

- **Jansen Slinger** (Institute for Organisation and Information, Utrecht University),
  **2009-09-18 09:39:50**
  Software vendors lack the perspective to develop software within a software ecosystem. The inability to function in a software ecosystem has already led to the demise of many software vendors, leading to loss of competition, intellectual property, and eventually jobs in the software industry. In this paper we present a research agenda on software ecosystems to study both the technical and the business aspects of software engineering in vibrant ecosystems. The results of such research enable software vendors to develop software that is adaptable to new business models and new markets, and to make strategic choices that help a software vendor to thrive in a software ecosystem.

- **A Sense of Community: A Research Agenda for Software Ecosystems**
  Jinger Jansen, Utrecht University - Anthony Finkelstein, University College London - Sjaak Brinkkemper, Utrecht University 18/09/2009
  In this paper a research agenda on software ecosystems is presented to study both the technical and the business aspects of software engineering in vibrant ecosystems. The results of such research enable software vendors to develop software that is adaptable to new business models and new markets, and to make strategic choices that help a software vendor to thrive in a software ecosystem.