

Capturing Design Rationale by Annotating E-mails

Yoshikiyo Kato, Koichi Hori

Department of Advanced Interdisciplinary Studies, University of Tokyo
4-6-1 Komaba, Meguro-ku, Tokyo 153-8904, Japan

and

Kohei Taketa

Department of Aeronautics and Astronautics, University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan

ABSTRACT

Development of a large and complex system, such as satellites, is a knowledge intensive task. Applying knowledge management to such a domain appears to lead to a more efficient development process. However, it has been reported that knowledge management suffers from capture bottleneck, i.e. lack of incentives for members of the community to invest time in sharing their resources, or knowledge. In this paper, we propose an approach to capture design rationale from e-mail communication. We describe the prototype system and analysis of applying the method to an e-mail archive of a satellite development project.

Keywords: Design Rationale, Capture Bottleneck, Annotation, Argumentation, Information Retrieval

1. INTRODUCTION

When developing large and complex systems like satellites, the development process becomes large and complex as well. As a result, it is almost impossible for a single person to cover the entire process in every details. In such a situation, communication between designers becomes critical in achieving a successful result. There are many cases where poor communication results in a failure of the system. For example, in the case of NASA's Mars Climate Orbiter [5], although the root cause of the failure was a mistake in the navigation software, the failure could have been avoided if a good communication was present between navigation team and development team; the navigation team did not conveyed their concerns to the development team even though they were aware of errors in the trajectory estimation of the spacecraft. In this paper, we address the problem of e-mail communication between developers, and show how we can apply the framework which the authors are proposing for capturing design rationale at low cost.

Nowadays, e-mail is widely used for communication. Especially mailing lists can be used by a group for announcements, discussion, etc. When mailing list is used for discussion which involves some kind of decision making, it is hard to track the argument and the reasons behind the decisions made. It is because standard e-mail systems only provide loose mechanism for organizing e-mails, i.e.

In-Reply-To references. It is often the case that several arguments proceed in parallel, and they are mixed in e-mails. Thus, finer grain structure is needed if we are to track discussions in e-mails more precisely.

To approach this problem, we focus on capturing argumentation structures in the communication as design rationale during the development process. Design rationale is the reasons, deliberations, or arguments on the design. By recording design rationale, designs become more understandable, and can be reflected on or reconsidered afterwards. The capture and use of design rationale has become one of the crucial aspects in design research [3]. However, it has been reported that there is a tremendous cost associated with capturing design rationales formally [1, 6, 8]. Similar problem is recognized as the 'capture bottleneck' in the domain of knowledge management [4]. The capture bottleneck problem in knowledge management refers to the lack of enough incentives to the users to invest time for sharing their knowledge or resources.

Motivated by the capture bottleneck problem, we are building a prototype system called IDIMS (Integrated Design Information Management System). The purpose of the system is to provide developers a unified repository of design information and to capture design rationale at low cost. As the use of computer is no more limited to analysis of artifacts and CAD, and widely used for communication and documentation, we believe that by actively utilizing all the electronically available information, we may reach to a solution to the capture bottleneck problem. Using the analogy to the recycling of wastes, we call the idea 'knowledge recycling'.

During development process of a large and complex system, huge amount of information is produced, whether it is formal or informal. Buried in pile of such information are *knowledge fragments*. A knowledge fragment is a piece of information which reflects deliberation, reasoning, or experience of developers. An argument behind a design decision, appearing in e-mail communications between developers, or a test report describing an anomaly observed during a test and its cause are examples of knowledge fragments. With additional cost on developers, we can collect knowledge fragments, put them together, and provide use-

ful information which supports critical decision-making in the development or operation of the system. Besides, accumulated knowledge fragments may serve not only a single project, but also projects in the future. We intend to implement and test the idea with IDIMS.

The rest of the paper is organized as follows. In Section 2, we describe the annotation scheme that we use for representing argumentation structure in Section 2. In Section 3, we explain the prototype system IDIMS, focusing on the features relevant to capture and utilization of design rationale from e-mail communication. Then, we present analysis of an e-mail archive by applying the proposed method in Section 4. Finally, we give conclusions in Section 5.

2. ANNOTATION SCHEME FOR REPRESENTING ARGUMENTATION STRUCTURE

To capture design rationale at low cost, we propose a scheme in which users are asked to annotate the design documents or e-mails to indicate ‘design issues’ or ‘design decisions’. IDIMS then processes documents and e-mails to collect annotations and stores them to Issue/Decision Repository.

I/D repository is intended to be used to capture argumentation perspective of design rationale [7]. We regard the annotation of issues and decisions as a light-weight version of design rationale representation, compared to existing representations such as IBIS [2] or QOC [8]. Instead of requiring users to build a separate design rationale structure, the method aims at capturing design rationale with minimal additional cost. We adopted simple representation to mitigate the cost of formalizing argumentation. It is reported that the cost associated with formalization is an inhibiting factor of the use of argumentation formalism to capture design rationale [1, 6, 8]. By adopting simple formalism and lessening the burden on users, we want to shed light on one side of capture bottleneck problem.

In representing argumentation structure, we use issues and decisions as basic constructs. *Issues* are problems or concerns about the design or implementation of the developing system. *Decisions* are resolutions to issues such as trade off between conflicting design parameters. Issues and decisions can be linked, and construct a directed graph. A link from an issue to a decision indicates that some kind of decision has been made on the issue. Inversely, a link from a decision to an issue would mean that a new concern has emerged on the decision that has already been made. A link from an issue to another issue would mean the latter is a sub-issue of the former, or the latter induced the former. A link from a decision to another decision indicates the latter is an auxiliary of the former (Table 1).

Link Type	Description
$I \rightarrow D$	A decision was made on an issue.
$D \rightarrow I$	A new issue arose on a decision which has already been made.
$I_1 \rightarrow I_2$	The latter issue (I_2) is a sub-issue of the former (I_1).
$D_1 \rightarrow D_2$	The latter decision (D_2) is an auxiliary decision to the former (D_1).

Table 1: The implications of links in the issue/decision structure.

3. INTEGRATED DESIGN INFORMATION MANAGEMENT SYSTEM (IDIMS)

Based on the concept of knowledge recycling, we are developing a prototype system called *Integrated Design Information Management System (IDIMS)*. IDIMS consists of four components, each of which corresponds to the information types it can handle: document manager, issue/decision manager, e-mail manager, functional-structure manager. IDIMS is a web-based system, and users access to the system via web. Configuration of the system is shown in Figure 1.

Document Repository

IDIMS has a document repository which stores the documents produced during the development process. It is implemented as a Java RMI service, and uses MySQL for storing data. A web interface is provided for the user to browse the repository. The user can add, search, and retrieve documents through the interface. Figure 2 is an example of a document being displayed in a web browser. To be displayed in web browsers, HTML pages are generated from XML documents with XSLT technology. User can also retrieve the original PDF document.

XML is the primary format of documents used in IDIMS. However, it has a mechanism to convert PDF documents to XML, and stores both the original PDF file and XML document in the repository. XML format was adopted to allow embedding design rationale tags in documents. IDIMS also accepts e-mails. E-mails are converted into XML format by E-mail Manager and stored into the repository. E-mail Manager is described later.

Issue/Decision Repository

Issue/decision (I/D) repository stores issue and decision annotations to documents. We implemented Issue/Decision Viewer, which is a tool for viewing issues and decision in the repository. Issue/Decision Viewer is a Windows application that connects to the I/D repository and visualize the I/D structure (Fig. 3). Users can register, search, and view issues and decisions in the database.

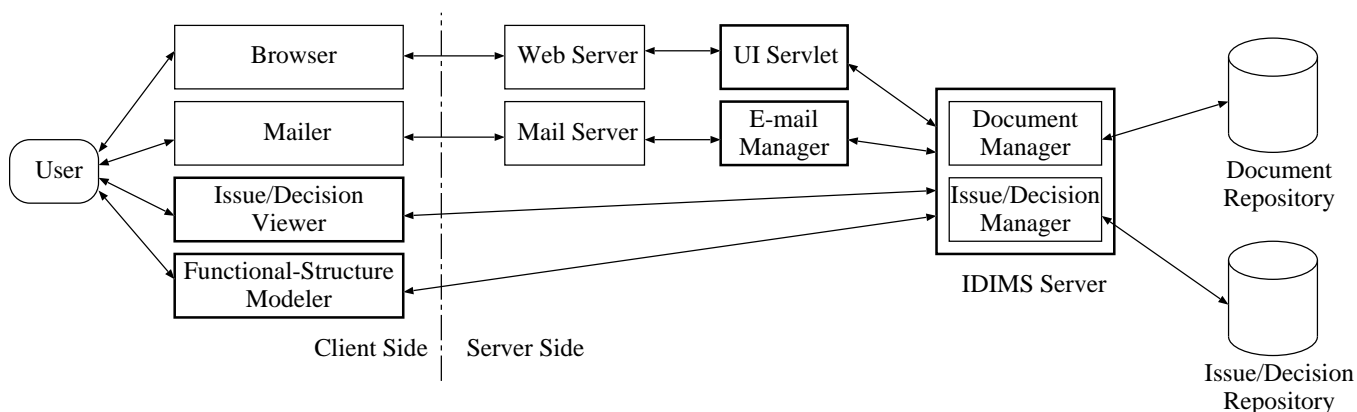


Figure 1: The system configuration of IDIMS. Components in the box with thick lines were built by the authors.

With this view, the argumentation structure would be readily visible to the user. User can register issues and decisions either alone to the database, or as annotation to a part of document.

For utilizing recorded argumentation structure, IDIMS provides the following features:

Listing unresolved issues. IDIMS provides a feature to list those threads of discussion which do not have decision. This feature was implemented in order to help users identify unresolved issues. By this feature, users can always return to the archive of e-mail discussion, and find those issues which have not been considered yet.

Thread-based search. IDIMS provides a thread-based search mechanism. It searches for similar threads in the Issue/Decision Repository based on a modification of TF/IDF ranking. It was modified so that a thread of discussion is treated as a document, instead of treating each e-mail message as a document.

E-mail Manager

E-mail manager processes annotated e-mail messages. E-mail manager is implemented as a mail filter. It is configured to receive mails sent to a designated mailing list address. Upon receiving an e-mail, the filter program transforms the message to an XML document, stores it to the document database. When the message is stored to the document database, annotations in the document are extracted and stored in the I/D repository. Each issue/decision is given a unique ID number when registered to the I/D repository.

Then, the filter program sends out the message to members of the mailing list. When sending out a message, filter program attaches a template for reply. This template cites the original message with citations annotated. Annotated citations in reply templates are given references to the annotation in the original message using the ID assigned. In

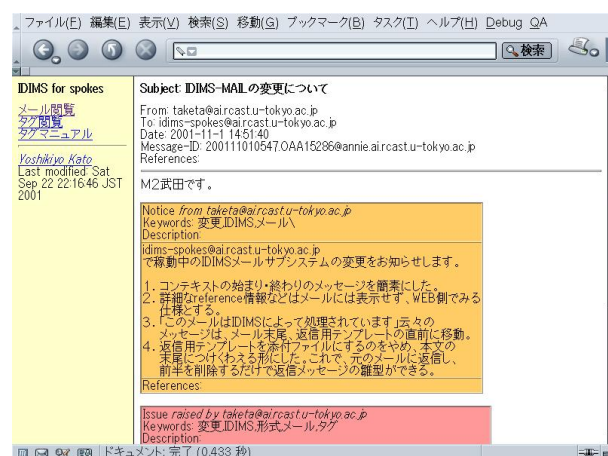


Figure 2: A document displayed in a web browser. Tagged block of text is shown in a box with a color.

this way, links between issues and decisions in e-mails are semi-automatically constructed, and threads of discussion are recorded.

4. UTILIZING ARGUMENTATION STRUCTURE OF E-MAIL COMMUNICATION

To illustrate how the method improves the communication and development process, we analyzed the log of a satellite project developers mailing list. One of the authors manually annotated about 700 e-mails from the log, and we got a set of argument structures. We counted the number of 'unresolved issues', which are those issues that do not have decisions associated with them, and categorized them. We compared two thread structures of the e-mails: the structure built from e-mails' In-Reply-To references and the structure built from Issue/Decision annotation.

During the analysis, we found several cases in the log which illustrate the communication problem. One of the

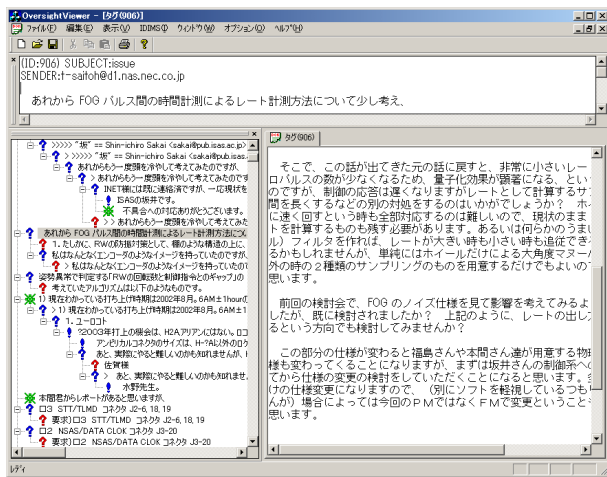


Figure 3: A screenshot of Issue/Decision Viewer.

problem found was that the same argument on an issue reappears, and it is not based on the decisions that have already been made. To avoid these kind of reappearing arguments, we propose thread-based searching mechanism. In an preliminary experiment, although thread-based search did not show significant improvement over keyword-based search compared in precision, thread-based search was more robust in finding similar threads, while keyword-based search was sensitive to the keyword selection.

The analysis shows that the proposed annotation scheme is promising in capturing design rationale and improving communication problems. However, further user studies are required to prove the cost-effectiveness of the method, which is the authors final goal.

The following sections describes the analysis of e-mail communication between developers of INDEX satellite. INDEX is a piggyback satellite under development at the Institute of Space and Astronautical Science (ISAS) of Japan. E-mails were taken for analysis from the developers mailing list archive. The authors annotated 679 e-mails with the annotation scheme described in Section 2.

Unresolved Issues

We hypothesized that issues which do not have explicit decision (*unresolved issues*) could become a problem afterwards. By using Issue/Decision Viewer’s capability to search for unresolved issues, we identified 84 unresolved issues. We categorized unresolved issues into five types as shown in Table 2. “No response” means that there is no corresponding decision for the issue in the e-mails analyzed. For “Unknown” type of issues, authors could not understand what the problem was. We count “no response” and “unknown” type of issues as net unresolved issues. Thus, the frequency of unresolved issues appearing in the mailing list becomes one in every eleven e-mails. We could not find a case in the archive where an unresolved issue

Type	Count
Decision exist in a different thread from the issue.	11
Issue was succeeded by another issue in other thread.	2
The issue seemed to be resolved, but the decision was not made explicit in mailing list.	9
No response	56
Unknown	6
Total	84

Table 2: Categorization of unresolved issues.

MTQ Magnetic torquer. An electromagnet used for altering the attitude of a satellite by generating electromagnetic field, which interacts with terrestrial magnetism and induces torque.

OBC On-Board Computer.

PCDYN A dynamics simulator which acts as an environment for the testing of the satellite attitude controller. It takes the controller’s outputs as its inputs, and provides its outputs to controller’s inputs.

Table 3: Acronyms used in the scenario examples.

later become the cause of a problem. It might be that those *unresolved issues* are not really left unresolved, and the decisions made are not reported with e-mail. However, it is important to explicitly representing decisions. Otherwise, there is always a risk of unresolved issues not being considered and leading to an failure.

Unattended Decisions

In analyzing the INDEX project mail archive, we have found several cases where an issue is raised on a matter, on which a decision has already been made. An example is shown in Fig. 4. Acronyms used in the example are explained in Table 3. Although in Thread 1 (Fig. 4(a)), C is stating on how to measure the output for MTQs, B raises an issue related to measurement of MTQ current in Thread 2 (Fig. 4(b)). This case illustrates how miscommunication or misunderstanding can happen. In this case, misunderstanding was corrected by another member and did not lead to a fatal consequence. However, it is possible that a misunderstanding happens on a critical matter, the project proceeds without it being corrected, and finally result in a failure.

One way to reduce these kind of misunderstandings or miscommunications is to improve accessibility to information such as design decisions and design rationales.

A (Issue): What are the pin numbers for the output to magnetic torquers(MTQ) (x, y, and z-axis)?

C (Decision): The pin assignment of P2 connector is as follows:

X-axis: 20 and 8

Y-axis: 21 and 9

Z-axis: 22 and 10

Note that you need to connect a register of resistance bigger than the torquer's resistance, when you measure the output without an actual torquer connected. Because the output voltage is the potential difference from the signal ground produced by a bridge circuit, measuring by an oscilloscope with a single probe could cause a short circuit. In such case, use two probes and read the potential difference between them.

(a) Thread 1: A is asking the pin assignment for the output to MTQs. C is responding to A, with a caution on how to measure the outputs without actual torquers connected.

B (Issue): We need to consider the following points:

1) Operating without load, PCODYN cannot get correct MTQ status, because it is produced by measuring the torquer's current.

2) It is possible to change the specification of the pseudo-sensor so that the PCODYN directly gets the OBC output. It takes about 3 hours for the change.

D (Response): I thought we have agreed upon using resistors in place of MTQs, when we discussed the testing system. If the dynamics simulator should directly get the command off the on-board software, there isn't much reason to use the dynamics simulator as it is the same as off-line simulation. If you are to evaluate the attitude control system, the test of MTQ driver should be included as well.

B (Decision): You're right. I keep forgetting things.

(b) Thread 2: B states a problem on measuring the MTQ current without load. D points out that it has been decided to use a load in place of actual MTQ when testing.

Figure 4: An example of raising an issue on a matter, on which a decision has already been made.

5. CONCLUSION

In this paper, we proposed a method to capture design rationale from e-mail communication. The method was implemented in IDIMS, a prototype system developed as a platform for knowledge recycling. Although the result of preliminary analysis is promising, we still need to test the method in a user study. We plan to conduct a user study in a student project to develop a small satellite.

ACKNOWLEDGMENT

We would like to thank Professor Hirobumi Saito (Institute of Space and Astronautical Science) for providing e-mail archive of the INDEX project. We would like to thank Yosuke Fukushima (National Space Development Agency of Japan) for introducing us to the INDEX project and the fruitful discussion.

References

- [1] E. J. Conklin and K. B. Yakemovic. A process-oriented approach to design rationale. *Human-Computer Interaction*, 6:357–391, 1991.
- [2] W. Kunz and H. W. J. Rittel. Issues as elements of information systems. Technical Report S-78-2, Institut für Grundlagen Der Planung I.A, Universität Stuttgart, 1970.
- [3] T. P. Moran and J. M. Carroll. *Design Rationale – Concepts, Techniques, and Use*, chapter 1 Overview of Design Rationale, pages 1–19. Computers, Cognition, and Work. Lawrence Erlbaum Associates, 1996.
- [4] E. Motta, S. B. Shum, and J. Domingue. Ontology-driven document enrichment: principles, tools and applications. *International Journal of Human Computer Studies*, 52:1071–1109, 2000.
- [5] MPIAT. Mars program independent assessment team summary report. Technical report, Mars Program Independent Assessment Team, NASA, March 2000.
- [6] F. M. Shipman and C. C. Marshall. Formality considered harmful: Experiences, emerging themes, and directions on the use of formal representations in interactive systems. *Computer-Supported Cooperative Work*, 8(4):333–352, 1999.
- [7] F. M. Shipman and R. J. McCall. Integrating different perspectives on design rationale: Supporting the emergence of design rationale from design communication. *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing*, 11(2):141–154, 1997.
- [8] S. B. Shum and N. Hammond. Argumentation-based design rationale – what use at what cost. *International Journal of Human-Computer Studies*, 40(4):603–652, 1994.