MORADOR

This is a project memo from the CAISOR-Morador activity concerning
Methodology Of Research And Dissemination Of its Results.

Erik Sandewall

The Methodology of Design Iteration for Systems-oriented Research in Computer Science

This project memo is published on the CAISOR website
at URL http://www.ida.liu.se/ext/caisor/pm-archive/morador/001/.

Related information can be obtained via the following WWW sites:

CAISOR website:	http://www.ida.liu.se/ext/caisor/
Morador open space:	http://www.ida.liu.se/ext/morador/
The author:	http://www.ida.liu.se/~erisa/

Date of Manuscript: 2006-04-07

Abstract

This memo addresses the methodology of research in systems-oriented computer science. We formulate the method of *design iteration* as a principled way of describing actual working practice in those research projects where large and complex software systems are built, and where a proposal for a software architecture is one of the major perceived results of the research. The iterative character of such research causes particular problems for the communication of its results. We argue that a new publication paradigm will be needed for effective communication between researchers that use design iteration, and propose a publication structure that should be adequate for this purpose.

Webpage for Discussion and Latest Version

Debate about the propositions in this memo is welcomed. A persistent webpage has been set up for it at

http://www.ida.liu.se/ext/caisor/pm-archive/morador/001/

Please refer to that page for an up-to-date account of discussion and for the latest version of the memo itself.

1 Methodology in systems computer science

1.1 The topic of this article

Some of the research in computer science addresses questions of the form 'what is the appropriate overall structure for software that is to perform tasks in C' for a designated class C. This article addresses the questions of research methodology and publication structure for research of that kind. The term 'architecture' is often used for the overall structure of the software, and I shall use it here in a way where it is taken to subsume both algorithmic control structure, dataflow structure, the structure of data within the computational processes and in the interfaces between them, and other similar aspects.

My primary concern is with research that strives to identify an architecture for a particular class of difficult problems, such as 'understanding natural language' or 'getting a vehicle to drive autonomously cross-country'. I am less concerned with research that proposes general software engineering techniques that are intended to be applicable across all kinds of problems, although maybe that can be considered as a higher-level design problem so that the discussion in this article may turn out to be applicable anyway. I shall not be concerned with such systems research where the overall architecture of the software system is already understood in the state of the art, and where the research is concerned with finding the best choice for a algorithm, subsystem, or other specific aspect of the design.

The phrase 'architectural studies in systems computer science' is appropriate for characterizing the topic being addressed. I would avoid 'architectural computer science' since research of this kind is not an area in itself; architectural studies co-exist with other research in several parts of our field.

Although the architecture of software systems is the topic being addressed, much of what will be said here can also be applied to other kinds of complex design creation, such as in the design of complex ontologies.

1.2 The structure of research results

Every research article is based on some assumptions about the answers to the following questions:

1. What is the form of research results in the area addressed here?
2. Within the framework defined in the first item, what are the particular results that are put forward in the article at hand?
3. What kind of evidence is needed in order to validate a proposed research result in this area?
4. What is the evidence for the results in the article at hand?
5. How do these results relate to other results (by other authors, or in other publications by the same author) in the same area?

The evaluation of the article, for example in the reviewing process, must also be based on assumptions concerning what makes a result 'interesting' and 'valuable'. In research areas with a well established methodology one does not have to worry about questions 1 and 3, since the answers to those questions are well established and accepted by everyone in the field. However, if the answers to questions 1 and 3 are not well understood in a branch of research, then there is a risk of intransparent answers for questions 2, 4, and 5.

I propose that architectural studies in systems computer science suffers from exactly that kind of weakness. Suppose that two research articles are given that describe two different 'architectures' for the same or similar tasks; suppose that their respective proposals are described in different terms, and you wonder, as a reader, whether these proposals are in fact equivalent and intertranslatable, or whether they are instead different and incompatible. The average article does not assist the reader in finding this out. I invite the reader to consider whether the articles that she is used to reading are based on clear answers to question 4, or to question 5, or even to question 2.

The methodological weakness for systems research in computer science, or at least this kind of systems research, may surface in other ways as well. The following may be a familiar scenario in the context of conference organization for a series of annual conferences in a particular field.

- Conference participants complain that the conferences have too many theoretical papers and not enough systems papers;
- The program committee answers that they would be more than happy to accept more systems papers, but they receive too few submissions of this kind, and those they obtain are not up to standards;
- Systems-building researchers in the same field complain that it is impossible to get their papers accepted although they have great results to report;

- Everyone agrees that the problem should be fixed at the next conference. This is implemented as a statement in the next call for papers to the effect that 'papers describing working systems' or 'papers based on the evaluation of actual implementations' are particularly welcome, but this does not have much effect.

If the reader has been part of this kind of discussions, or if he recognizes the problem of formulating and validating results in architectural software research, then the continuation of this article should be of interest.

1.3 Classical methodology advice

In one line of proposals for the problems described above, systems researchers are advised to take 'standard' methodology seriously. Each article should describe clearly what is the architecture they propose, it should specify in what ways the architecture is claimed to be superior to other systems for the same purpose that have been reported previously, and it should give adequate evidence for that claim. The message is 'there is nothing special with systems building research, or architectural systems research, so you should just learn to do research properly'.

There are however a number of problems with this seemingly reasonable advise. An architecture is often a quite complex object involving a large number of weakly interdependent design decisions. Comparing two architectures (your own, and someone else's) requires comparing two systems that differ in many ways; it is hardly possible to relate differences e.g. in measured performance to one or a few of all those design aspects. Furthermore, it is not likely that earlier systems have been described in such terms that those comparisons can even be contemplated.

In order to address this problem properly it is necessary to step back and to first address the question of methodology per se: what is the natural discovery process in systems computer science?

2 Design iteration

Consider the following scenario. A researcher R decides to address the problem of how to properly design software systems that perform a particular target task T. We assume that T is quite difficult, so that no one has been able to do it before, although there may be a number of competitors in the field that try it concurrently with R.

R sets out by defining a *design hypothesis*: an idea about how the system doing T should be organized. A large part of the work in the project then consists of extending the design hypothesis and making it more detailed and concrete.

This is done concurrently with implementation. R builds a 'prototype' or 'experimental' software system that is intended to correspond to the design hypothesis. While doing so, R will sometimes arrive at points where she sees that some aspect of the initial design hypothesis was wrong, and needs to be changed. In principle, it would then be appropriate to discard the implementation at hand and start over. However, if the design hypothesis is

large and the software that has been implemented so far is correspondingly fairly large, then it may not be reasonable to discard the entire software system every time an additional design revision has been made. Instead, R considers that it is better to continue developing the actual software system *although* her current design hypothesis DH(t) at time t differs not only from the initial design hypothesis DH(0), but also from the current implementation CI(t) at time t. This is reasonable as long as the difference between DH(t) and CI(t) is relatively small, so that enough can be learnt by pursuing CI in spite of its distance from DH.

Necessarily, due to diminishing returns, there comes a time u when the difference between DH(u) and CI(u) is so large that it is better to discard CI and start over, implementing a new system based on DH(u), that is, the researcher's present best understanding about how the system should be built. This starts another cycle in an iterative process of *design iteration*. Hopefully this iterative process will eventually terminate. This occurs if DH(0) = DH(u) = CI(u) at a time u when a complete system exists, that is, if DH(0) is a fixpoint under the implementation operator.

This is all well if the time from 0 to u can be measured in weeks or months, and if only a modest number of cycles is required in order to reach the fixpoint. In this case the final outcome of this process can be reported as the result of the research project within one to three years, for example through a Ph.D. thesis.

A major problem arises, however, if it is necessary to report the results of the work at the end of one of the iterations but without having reached a fixpoint. This is not an uncommon situation: the time when one generation of system implementation has been completed and the software is ready to be discarded without convergence, is quite likely to also be a time when the Ph.D. student that did the work has to write her or his Ph.D. thesis. What should she then write about? The initial design hypothesis DH(0) that has now been abandoned? The current design hypothesis DH(u) that is now hopelessly different from the current implementation CI(u), and for which no more accurate implementation exists? The current implementation CI(u) that is now known to be inadequate for the given task T?

An additional aspect that is also important for the scenario, is that the time of proceeding from one cycle to the next will also be a time for comparing the experience from one's own project with the reported results from others that have addressed the same task or related ones. In other words, the time between successive cycles is a natural time for research communication, including both the publication of one's own result, and digesting the results of others. The question "what should one write about at the end of the design cycle?" is therefore essential, not only for the individual author of the research article and his or her advisor; it is also a question concerning how the field should organize its own internal communication.

3 Identifying the results of nonconverged design iteration

The purpose of publication must always be to communicate. In terms of the scenario that has just been described, one must therefore ask what is

appropriate to communicate, between researchers, at a time u where one cycle of design iteration has been finished without convergence, that is, when the major outcome for the researcher is additional insight about what the next generation of implementation ought to be like.

A possible answer may be not to bother, and to only publish definite results with long-lasting relevance, which in this case would mean to make *no* publications for nonconverged designs. However, this position misses two important points: (1) long-term research on complex system designs must frequently work with schedules that last over many years, due to the number of design cycles and to their durations, and (2) there is a legitimate need, from the point of view of sound methodology, to have communication between projects with similar goals, even at the times when the design has not yet converged([1]). Furthermore, the methodological acuteness in this kind of research ought to benefit from the peer review and the public scrutiny of results that is regularly associated with scientific publication.

3.1 Comparing notes as a communication paradigm

In order to get another perspective on the communication issue, let us shift it for a moment from communication by publication, to communication by dialogue. Consider the following situation: two researchers have embarked on the same development task at the same time, and after having completed one or two development cycles they compare their current design hypotheses as well as the steps that led up to them. If these design hypotheses are similar and have converged, then one has obtained two independent verifications of the same result. However, what would be rational behavior in the more frequent cases where their design hypotheses have not yet converged, and their experiences and their current design hypotheses differ significantly?

If the classical point of view is applied to this situation with two researchers, the advise becomes that since their respective results at this point are inconclusive they should be ignored; these researchers should proceed until each of them has obtained stable, converged designs. Then they should evaluate the two designs, individually or jointly, in order to determine which one is best, or what is the best use for each of them.

Such a way of proceeding would not be reasonable, of course. There must be many things that are worth talking about between these two researchers as they exchange their experiences, and what they learn can be useful when they formulate the initial design hypothesis for the next design iteration. I propose that this simple observation should be generalized to the level of scientific publication: *the main purpose of scientific publication from architectural projects in computer science should be to exchange design experience that is useful for researchers that prepare to build a particular kind of system, either for an additional cycle of design iteration, or in order to use the system in a wider context.*

Adoption of this principle does not of course preclude that the same publications can be of interest for other readers and for other purposes as well,

[1] Notice, by the way, that if there is a norm that only definite results can be published, then it may easily happen that nonconverged designs are presented as being more definite and better converged than they actually are.

but it is useful as a basis for discussing what is an appropriate structure for the publications.

3.2 Nonstandard publications

The scientific community today is very focussed on the standardized article: standardized with respect to both style, size, and type of contents. In some environments this may be a rule without appeal, namely, if accepted articles in quality journals and conferences are the only things that are read, cited, and given credit. However, a plausible diagnosis for the conference illness that was described above ('please let us have more systems papers and fewer theory papers') might be that the traditional article format is not well suited for communicating the results of architectural, systems-oriented research. Let us therefore add some degrees of freedom by allowing for the possibility that the publication and communication of research may use other forms and styles.

There are many possibilities with respect to form. Perhaps we should allow for fewer, longer publications. This causes problems if research is evaluated by counting the number of titles (why not count the number of pages, if you insist on being quantitative?), but let us leave that problem aside. Maybe we could use publications that are formed as a cluster of many short 'notes' of a few pages each that form a kind of hyperbook?

The topic should not be restricted to the use of written text. In some parts of chemistry, research articles are usually only of interest if the author has submitted the results in tabular form into a database that is operated jointly by the research area. It is easy to imagine how at some point it is the database contribution that becomes the essential result, and the 'article' or 'paper' becomes a wrapper.

Publications may legitimitely occur in several layers. In our field we are only used to ordinary articles and survey articles (besides book reviews, etc). Consider, however, the practice in clinical medical research where researchers regularly produce and publish 'studies' reporting observations for a certain number of patients from their own practice or otherwise within reach. These 'studies' constitute one layer of publication. The next layer is obtained when a sufficient number of studies have been performed, and an author or group of authors collects studies from several sources, analyzes them, and identifies whatever conclusions can be drawn. The articles in the first layer serve as the base information and the evidence for the results that are presented in the second-layer publication. The readership for the second layer is of course much larger than for the first layer, but it is essential that even the first layer articles are publicly available so that they are open for inspection, and so that their results can be reused for additional second-layer articles when the occasion arises.

Our field does not have much that corresponds to these publication practices. Maybe we should. Why has not the field of knowledge representation established a mechanism where contributions to individual parts of a large ontology can be submitted, peer reviewed, incorporated into the universal ontological structure, and used by others? And why don't we require that software whose existence and properties are reported in articles in our field, should be validated by depositing both source code, test suites, and logs of system execution in a way where it can be inspected in subsequent studies?

For that matter, why do we have so many articles where an author or group of authors describe their own system, and so few articles where the authors compare and analyze several systems in an insightful way? Is it because the first-level presentations of the individual systems do not provide the information that would be required in order to make the second-layer publication possible? - or is it because articles of that kind would not be given scientific credit? Other explanations may be also be possible: "not encouraged by the funding agencies", "such work can not be done by graduate students, and they are the ones that do the research here", "not as fun as building your own system, and a sure way to aggravate your peers".

There are remedies for all of these objections if we just decide to, of course. For example, with respect to the aggravation, one way to deal with that is for the originators of several systems to write the second-layer article jointly.

4 Proposed publication structure

After this discussiono of alternatives we can return to the question of communication and publication for our own field. The scenario with two researchers that compare notes when they are at the end of their respective design cycles is suggestive for what kind of information should be communicated, but it must be complemented by another aspect, namely structure. A dialogue between two persons may organize the topic of discourse as it goes along, but in a publication situation with one-way, one-to-many communication it is important to have some structure for the package of information that is to be presented to the readers.

4.1 Primary publications

The question that was posed towards the end of section 2 concerned what the researcher should write about at the end of a design cycle for a nonconverged design, which are then the primary publications from the project at that time. The proposed answer to this question is: *Write two documents, one of which has the character of an article, whereas the other one provides evidence for the propositions that are made in the first one.* I shall refer to these as the design analysis article and the realization report, respectively. Both will be called *publications*; I use this term for its common-sense meaning as *a document that has been made public*, without any restriction to peer-reviewed works.

The **realization report** should have the following major contents:

- A concise description of the premise, that is, the system design as it was conceived at the outset of the realization.
- A discussion of the premise.
- A description of how the realization was done, that is, the experimental implementation of the premise.
- A description of the results, namely, the additions and modifications of the premise design that were done in the course of the realization, as well as the limitations to the applicability of the premise design that were identified during the realization.

Notice that a discussion of previous work by others is not included in this structure; the report is entirely focussed on the work being reported. Notice also the strict separation between the description of the design premise and the discussion about it. The realization and results sections should be based only on the premise section, whereas the purpose of the discussion section is to motivate why this particular premise was chosen as the starting point of the work.

The purpose of the **design analysis article**, on the other hand, is to communicate the results of the work to fellow scientists, in particular those that wish to build their own system for the same task or a similar one. The article should therefore describe and motivate a design that is a likely candidate for the design premise in a new project, but it should also give an insight into the possible alternatives in the design.

For this purpose, a design analysis article should be based on one specific realization report. The step from a design report to an analysis article should involve a review of the available literature and of the results from other research, including both theoretical research and other experimental system realizations. It should also of course include a revision of (or an alternative to) the design premise of the underlying project in view of the results that were obtained there. Finally, it ought to discuss what are the implications of the work being reported on the conclusions that have been reported in other research on the same architectural topic.

The realization report and the design analysis article play distinct roles and apply to the situation where one iteration of a design cycle has been completed, and the next one is being prepared. This invites however the question for how this cyclic process gets started. Is there supposed to be any document at the beginning of the first cycle?

My proposal is to identify this as a third type of document, which may be called an **initial design proposal**. Like the design analysis article it should contain a discussion of issues, but like the realization report it should not qualify for journal or journal-like publication. Since cyclically, every realization report should refer back to at least one preceding design analysis, the realization report of the first cycle should refer back to the initial design proposal.

4.2 The state-of-the-art article

The initial design proposal, the realization report, and the design analysis article are the ones that directly reflect the design iteration process. In addition, there will be a need for publications of a more overriding character, as well as documents that provide even more detail than the realization report.

In particular, I propose to identify the **state of the art article** as one that gives a balanced view of the latest results with respect to software architectures for a particular kind of system, identifying those aspects of the overall design where there is general agreement, those where there are alternative proposals all of which merit further consideration, and at what points there is simply a lack of adequate solutions.

4.3 The realization diary

The realization report should describe those design changes that have occurred during one cycle of design iteration, to the extent that those changes have a sufficient significance. Although this report is likely to be written towards the end of the cycle, it must be based on notes that are made continously during the work in order to be reliable. A **diary** or **journal**([2]) that is maintained throughout the realization work is therefore an important part of the methodology.

It seems reasonable that the realization diary should also be made available to peers on demand, so that the background for the statements in the realization report is also inspectable. This may be arranged either by a practice where the realization diary is posted on the project website, or by a practice where access can be requested on a person-to-person basis. In the former case the diary is to be considered as a 'publication' so that it can be cited as such; in the latter case it will be seen as an 'unpublished document'.

5 Structure within publications

We turn now to the structure within the publications, in particular for the realization reports. To the extent that one can formulate a strict methodology for architectural systems research, it will be manifested the most strongly in these.

5.1 Structure of realization reports

Realization reports should be written with three kinds of readers in mind: authors of a design analysis, scientists that prepare to build a similar kind of system, and readers of a design-analysis article that have become curious about a specific point in the article and wish to check it out more carefully. This means that the readership is not going to be very large; it would not for example make sense to include such reports in printed journal issues. It is important however that design-cycle reports are *publicly available* for an extended period of time, so that the reader of a design-analysis or state-of-the-art article is able to check out its sources. It is in that sense that they ought to be considered as 'publications'.

What should be the structure of a realization report? In section 4 we proposed what should be its major contents; the following is a more detailed proposal. Quite possibly it will have to be revised after we have had experience with using this way of writing in a number of projects.

1. A brief description of the premise, i.e. the design hypothesis DH(0) at the beginning of the design cycle being reported.
2. A concise description of DH(u), the current design hypothesis at the end of the cycle, with some more details than for DH(0) but still without going into details. The description of DH(u) should point out what are the major differences between DH(0) and DH(u), since this is the incremental information that has been obtained from the

[2] A journal in the sense of 'an account of day-to-day events' (Webster).

point of view of the researcher herself, but anyway the description of DH(u) should be self-contained.

3. A discussion of the premise and of the concluding design hypothesis DH(u).
4. A characterization of major milestones in the development history, if any can be discerned. For example, if there was a major redesign in the middle of the development period, so that it is really two successive design cycles that are being reported, then this should be explained.
5. A brief description of CI(u), the current implementation at the end of the cycle, describing its major properties and how they differ from DH(u).
6. A chronology of the *design evolution* in terms of *design choices* and *design changes*. The chronology should be precise about the character of, and the reasons for each choice and change.
7. Optionally, a brief description of the applications that the system has been tested with in the course of its implementation. This is relevant in order to provide the context for the accounts of the design evolution and the design features.

The four items that were identified in the previous section correspond to item 1, item 3, items 4, 5, and 7 together, and item 6, respectively. Item 2 can be seen as a kind of summary that will give the reader an overview of the situation before going into the details. Notice that it is section 6 and not section 2 that are to be considered as the major results of the work, since any subsequent analysis in later publications must be based on the details in section 6, and it should not merely pick up the overview in section 2.

As always is appropriate to have a final section in the document summarizing what has been said.

Each realization report ought to be accompanied by an overview of underlying theory and of previous work on the same topic. Contemporary conventions would dictate that this background material be included in the report itself, but there are also advantages with keeping them separate. The background text may apply to several successive realization reports for successive iterations, and in this case there is no point with including similar material in all of them. The background is not only needed for the realization report; it is also important when the project is presented to various interested parties (sponsors, industry cooperation), and it may be needed for internal information within a large project. Furthermore, to the extent that the background text is amended from time to time, these amendments do not necessarily co-occur with the turn of the design cycle.

There is a traditional view that research articles should be 'self contained' in the interest of the readers. This may have been more important in the days of paper-print technology, but when a research group organizes and maintains its publications on its own website, it makes more sense to emphasize modularity in the structure of those publications and to work with cross-links between the publications instead of duplicating similar contents.

5.2 Criteria for design amendments

Design changes may be due to insights that are gained when details are successively added to the system itself, but they may also be due to the needs of emerging applications of that system that are attempted concurrently with the implementation of the system as such. In either case the realization diary should record the reasons for the change, which means that emerging applications have to be explained as well.

A realization diary can certainly be written in free-format style, as a list of individual updates each of which is explained using a paragraph in natural language. Additional structure can be added, however, if we can identify and classify a set of schemas for when it is appropriate to change a design, and for what forms a design-change can take. The following are some first thoughts on those topics.

The desire to avoid redundant concepts or constructs seems to provide one such schema. If, during the design refinement process, one arrives at a point where it becomes (locally) necessary to extend the design in such a way that the 'same' thing is implemented in different ways in different parts of the overall system, then one has reached an undesirable point. If one can then also see that another choice at an earlier point in the design process would have avoided the duplication, then a change of design hypothesis is warranted.

Some discrimination is warranted in using this schema, however, since there are cases where redundancy is tolerable and can be retained from one design cycle to the next, namely, when it is not critical to the overall design hypothesis. Consider, for example, a question-answering system that uses natural language both for acquisition of its information base, and as a query language. Suppose one has a choice of either using two separate, existing natural-language systems for information acquisition and for queries, or of using one single system which however requires a certain amount of work in order to handle the dual task. Suppose further that that choice does not have any bearing on the reasoning tasks that are considered to be the significant ones in the project. In such a case one may of course choose to stay with the two separate subsystems.

The quest for efficient execution of the software provides another schema for amendment of the design. If the implementor sees that she has painted herself into a corner where only a solution with unsatisfactory performance is possible, then again a change of design hypothesis is warranted. These two types of undesirability are of course related, since duplication of features may arise exactly in order to 'rescue' performance.

In another perspective, one may wish to characterize different amendments with respect to how they modify the design hypothesis at hand. Three alternatives come to mind at once, namely, *replacement of a component*, *reshuffling of structure*, and *installation of a handle*. A component, in this case, can be either an algorithm, a choice of data structure, or a choice of descriptive data that occurs as a constant in the program, to mention just a few examples. Reshuffling of structure includes, for example, a change in the order in which different things are done. Installation of a handle, finally, can consist (in the case of a program) of embedding an existing program component in a `case` statement where it becomes the `else` branch, and where other branches can easily be added to serve specific needs. The

observation that a particular handle was warranted in a particular part of the design is then an example of a result from the design process that may sometimes merit being recorded, together with the reasons why that design decision was taken.

Schemas, such as these, for the character of design amendments and for their respective rationale are motivated by the methodology of design iteration as a way of doing research on complex, innovative designs. To the extent that they are applicable they can help the researcher in organizing his work and its results, and they serve as a framework for a potentially large number of small-scale intermediate results within the project.

5.3 The realization diary

Ideally one would like the researcher to record each amendment to the design in clear form exactly when the decision is made. In practice they do not always occur in such a crisp way: the researcher may have a fuzzy perception of the need, resulting in a first attempt at the design change, and then iterate for a while until the design revision is in place, either in the design hypothesis or in the current implementation, or both. It appears that design iteration is a hierarchical process, where a very local design development is made in a subordinate design iteration that was able to cycle to convergence and that was then included as an element in the higher-level design development.

The details of such a rapid, local design loop are not likely to be of interest to anyone, but the question is then how one is going to capture the higher-level design decisions effectively. One possibility is to begin with a **laboratory diary** where the researcher makes notes of his or her work on a continuous basis. The researcher will then reconstruct the realization diary by cleaning up the lab diary and removing irrelevant detail that is due to very short-term design iteration.

There is also another thing that needs to be done in the step from laboratory diary to realization diary, namely, filling in the gaps. In practice it is difficult to bring yourself to putting everything into a laboratory diary([3]), because in some situations you wish to focus entirely on the task at hand, without also having to pay attention to making notes. The resulting gaps in the notes can be filled when the realization diary is reconstructed.

One can think of several ways of capturing a log of successive design amendments in a project; what was described above was just one possibility. It is difficult however to imagine a fully automatic method, and in practice the preparation of the log of amendments will require an additional effort by the researcher.

This is in itself an obstacle. If at some point it has become standard practice in this kind of research to maintain a lab diary and to derive the realization diary from it, then things will be different, but at present there is no particular incentive to the researcher for doing this, except if she is interested in the methodology issue per se. Furthermore, the realization diary and even more the lab diary will record a number of decisions that later turned out to be wrong. Although these decisions may have been very reasonable at the

[3]This statement is based on personal experience; the present author has maintained a laboratory diary for his own work since several years.

time they were taken, they can anyway easily be perceived as mistakes when others read about them. The resulting embarrassment factor can cause the researcher to either be so vague in the diary that the real development is hidden, or to abstain altogether from preparing the diary.

The use of diaries for the establishment of priority of results may be a complicated issue. It may work both ways: publicly available logs of system development may conceivably be used to claim priority, but they may also turn out to be ways of 'leaking' results without obtaining proper credit, or of 'stealing' results from colleagues by putting them into your own logs. The overall publication system should be set up with safeguards so that this does not become a problem.

5.4 The state of the art article

A state of the art article may be authored by a researcher with a strong background from research in the area being surveyed, but it would be even more interesting to have joint articles where the originators of several systems with similar goals discuss each others' systems as well as their own after having carefully studied each others' realization reports and diaries, as well as other information about each others' systems. Such joint analysis efforts could be very interesting program items at conferences and workshops, but the written presentation would have a strong advantage since its authors can include references into the underlying documents in order to support their arguments, and since its readers are able to check out the support by linking into the archived documents as well. It would help, in order to make this practicable, if the diaries can be indexed e.g. by assigning individual numbers or other identifiers to the successive design amendments that are reported in them.

6 A philosophical aspect

The proposals that have been made in this article make the assumption that research results in this area ought to be validated by an account of how they were developed. This means that one departs from the notion that new systems and system designs are to be tested experimentally, by formulating hypotheses about the design and its relation to alternative designs and testing those hypotheses in reproducible experiments. Instead of separating the implementation of the system from its experimental and reproducible validation, one considers that both the result and the validation of the result emerges from the research and development process. This may be controversial.

7 Introducing the new paradigm

A change of research paradigm does not happen overnight; a change of publication scheme takes time and patience as well. Some specific difficulties have been discussed above.

The proposals in this memo concern changes both in the paradigm for research on the architectures of complex software systems, and in the styles of

publication that are required for communicating such results. The change in publication style must be done on the community level. An individual researcher that tries to do it on his own or her own runs several risks. If he tries to publish in existing journals or conferences then the articles will run a risk of being rejected due to the use of unfamiliar methodology. If, on the other hand, the author chooses to self-publish the article using an on-line repository then the career may be damaged by a lack of refereed publications.

For these reasons, the new approaches that have been described here can best happen if one sets up new structures for scientific communication that are adapted to these approaches. It is a matter of choice whether these new communication structures are to be called 'journals', 'archives', 'transactions', or by some other term; the important thing is that they require a new set of expectations from authors, reviewers, and readers alike, as well as new ways of structuring the documents and the relations between the documents.