How to assign cost to "Avoidable Requirements Creep"

A step towards the waterfall's agilization

Stefan Pühl
Dell GmbH
Frankfurt, Germany
stefan_puehl@dell.com

Ralf Fahney
Fahney Anforderungsingenieurwesen GmbH
Zurich, Switzerland
rf@fahney.com

Abstract— **Scope creep is a major risk in fixed price projects. It has been suggested to distinguish between so-called "avoidable creep" and "unavoidable creep" where "avoidable creep" results from stopping requirements engineering (RE) effort too early e.g. for cost-saving reasons. However, no suggestion has been made how to assign cost to "avoidable creep" to quantify the consequence of too early stopped RE effort and to get that type of creep managed or even considerably reduced. From our experience in real and large fixed price projects, we derived a suggestion for solving this problem as soon as possible. The suggested solution has not yet been tried out in practice. However, with this paper we would like to begin a discussion about the suggested solution, and whether it could solve the problem of valuating "avoidable creep", could thereby reduce both the customer's and the supplier's risk in large fixed price projects and, as a side-effect, serves as a step in converging pure waterfall and pure agile procedures.**

Keywords – change request; fixed price contract; requirements management; requirements change; release management

I. INTRODUCTION

Scope creep is a major risk in fixed price projects, especially for the supplier, but also for the customer. This risk is even more important in large projects of dozen millions of EUR size. Berry et al [1] suggest distinguishing between "avoidable creep" and "unavoidable creep". They define "avoidable creep" to be creep caused by prematurely ending requirements engineering effort. This means requirements engineering efforts having been stopped before requirements were specified in the necessary detail. Such a creep usually occurs if for instance management decides to cut work on requirements due to cost-saving reasons. Berry et al define "unavoidable creep" to be creep "[…] that comes from just not knowing what the system is supposed to do […]". From our understanding, this definition includes e.g. unknown interrelationships between components or working with unknown technologies. Berry et al give a criterion when to stop requirements engineering effort to avoid "avoidable creep". From our knowledge, no suggestion has been made how to assign cost to "avoidable creep" to enforce making management concerned of that type of creep.

From our experience in large fixed price IT projects, we perceive not only considerable extension of requirements effort and requirements rework due to "avoidable creep" but also considerable increase of implementation effort due to that type of creep (which is nothing else than a confirmation of what Berry et al. describe in [1]). In our case, the customers of such projects are large, globally operating groups. But the problem may apply to small and medium businesses as well. In our case, the customers usually commission a general contractor for the implementation of the new IT system. In some cases, the general contractor commissions subcontractors for implementation, for instance due to specialized know-how possessed by the subcontractors. Other delivery models exist but have not been explained in this paper due to space constraints.

From our experience with such projects, we motivate and raise the question how to manage, reduce and maybe even avoid "avoidable creep". We answer it by suggesting applying change request management not only to the implementation but to the system requirements specification (SRS) as well. And we suggest applying change request management to the SRS as soon as possible in the project.

From our perception, by now authors focused on small teams (e.g. [2] has just the Crystal Clear version elaborated to a considerable extent), pure organizational aspects of how to get waterfall and agile procedures aligned to each other (e.g. [3] and [4]) or consider cost just as an outcome of projects (e.g. [1]). Authors only state that scope management is an issue. They did not suggest how to make management understand the impact of scope creep not only conceptionally but tangibly. This problem is what we address in our paper. Management thinks in terms of cost. We suggest how to make scope creep cost perceiveable to management at the most early point in time possible.

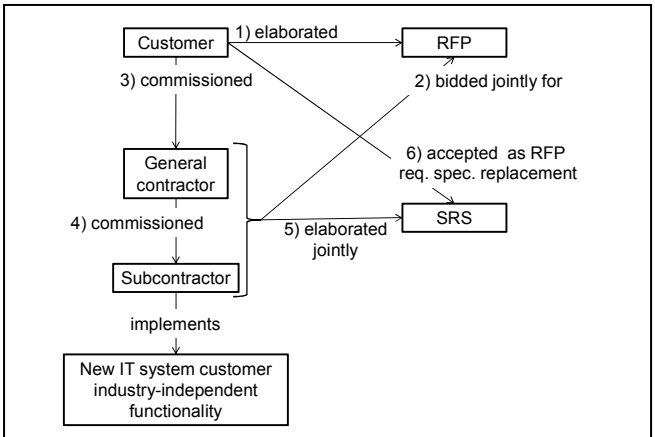

Figure 1. Relationships between involved parties and elaborated results

We assume that the suggested solution will result in some implications during project acquisition and contract negotiations, e.g. possible difficulties in selling the project and explaining to the customer the problem and the benefits of the suggested solution. However, we think that by thorough communication the risk of being misunderstood could be reduced considerably and maybe even replaced by a sound understanding of jointly proceeding on safe ground, a true win-win-situation.

In section II, we describe in detail the type of projects we are reporting about. Section III states the problem. Our suggested solution is described in section IV, followed by an explanation of benefits to the customer in section V. In section VI we motivate and suggest topics for future research. Finally, the conclusion in section VII summarizes what we achieved by now in addressing the problem, and what we were not able to achieve.

We think the suggested solution serves as a step in converging pure waterfall and pure agile procedures.

II. THE TYPICAL PROJECT

A. How large fixed projects usually are set up and planned

The experience we are reporting about here is about large and real projects using fixed price approaches and needing a certain amount of outsourcing. Because it is about real fixed projects, we have to mask most of the technical information, especially the name of the customers and the customers' industries. The problem that we will describe later, the suggested solution, and the conclusion however still hold.

The customers are large, mostly globally operating groups with revenue of over 10 or even 20 billion EUR and some 30,000 to 60,000 employees. They want to replace legacy IT systems by new developments to resolve problems like increasing inflexibility in terms of adaption to new market needs, maintainability, usability, etc. Such a customer elaborates a request for proposal (RFP) of several hundreds of pages size which becomes basis for a tender process for a fixed price commissioning of a general contractor to implement the new IT system.

The general contractor then agrees with subcontractors upon a fixed price contract basing upon the RFP and defining a work split. All parties (the customer, the general contractor and the subcontractors) agree that the RFP requirements section needs refinement (in general and between the suppliers) to arrive at requirements having an implementable level of detail. They agree to elaborate a detailed system requirements specification (SRS) which, provided the customer accepts the SRS, shall replace the RFP requirements specification section completely as the fixed price contract base. Fig. 1 shows the relationships described so far.

In parallel to the SRS elaboration, implementation starts with typically a proof of concept and some three or four construction iterations, each of them lasting a few months.

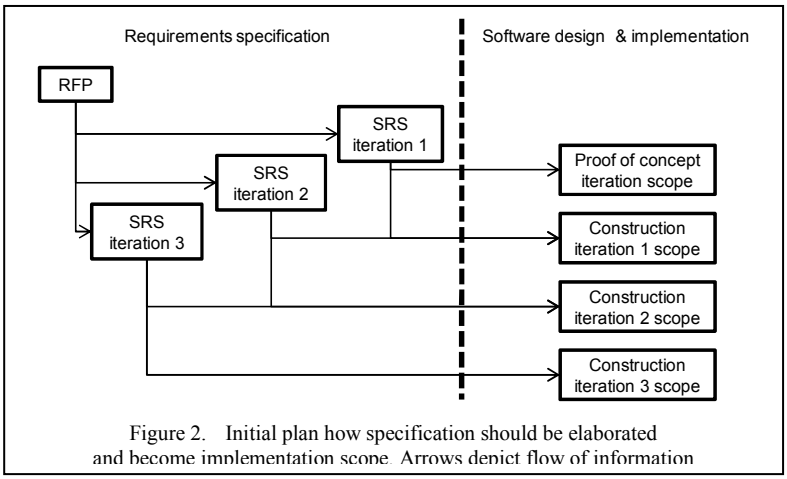

Figure 2. Initial plan how specification should be elaborated and become implementation scope. Arrows depict flow of information

The general contractor initially plans (Fig. 2) to elaborate the SRS in three iterations in a way that the proof of concept iteration could start based upon the first SRS iteration and all construction iterations could be planned and implemented completely based upon the finalized and agreed upon SRS. The general contractor and the subcontractors agree to elaborate the SRS from the end users' and business process perspective. This perspective is independent from the work split agreed between the general contractor and the subcontractors. For instance, the new IT system should allow for creation of two types of customers: Those with customer industry-specific attributes and those without such attributes. For the second type of customers, the new IT system should store industry-independent attributes only like name, address, birth date, hobbies etc. In the new IT system, both types of customer should be created by using the same self-service user interfaces (UI). One of the subcontractors specifies the requirements for the creation of customers. Hence, this subcontractor specifies that UI with customer industry-specific attributes even though according to the work split customer industry-specific data implementation might not be up to this subcontractor but to the general contractor or another subcontractor.

The plan of payments agreed upon between the parties is quite simple. The fixed price payment is distributed equally over the following events: Contract signing, RFP requirements specification section replacement, proof of concept iteration acceptance, one payment for each construction iteration, and one payment upon start of production of the new IT system.

B. How such large fixed price projects usually develop

We have made the following observations about such kind of projects:

1) The effort needed to elaborate the detailed SRS usually turns out to be much higher than initially estimated. We observe that many more SRS iterations are needed to arrive at an SRS that the customer is willing to accept as the RFP requirements section replacement. The SRS artifacts list initially elaborated needs to be extended several times. Amazingly, scope as such changes very rarely (not many change requests) as well as very little (small size of change requests). This is why we look at this phenomenon as being

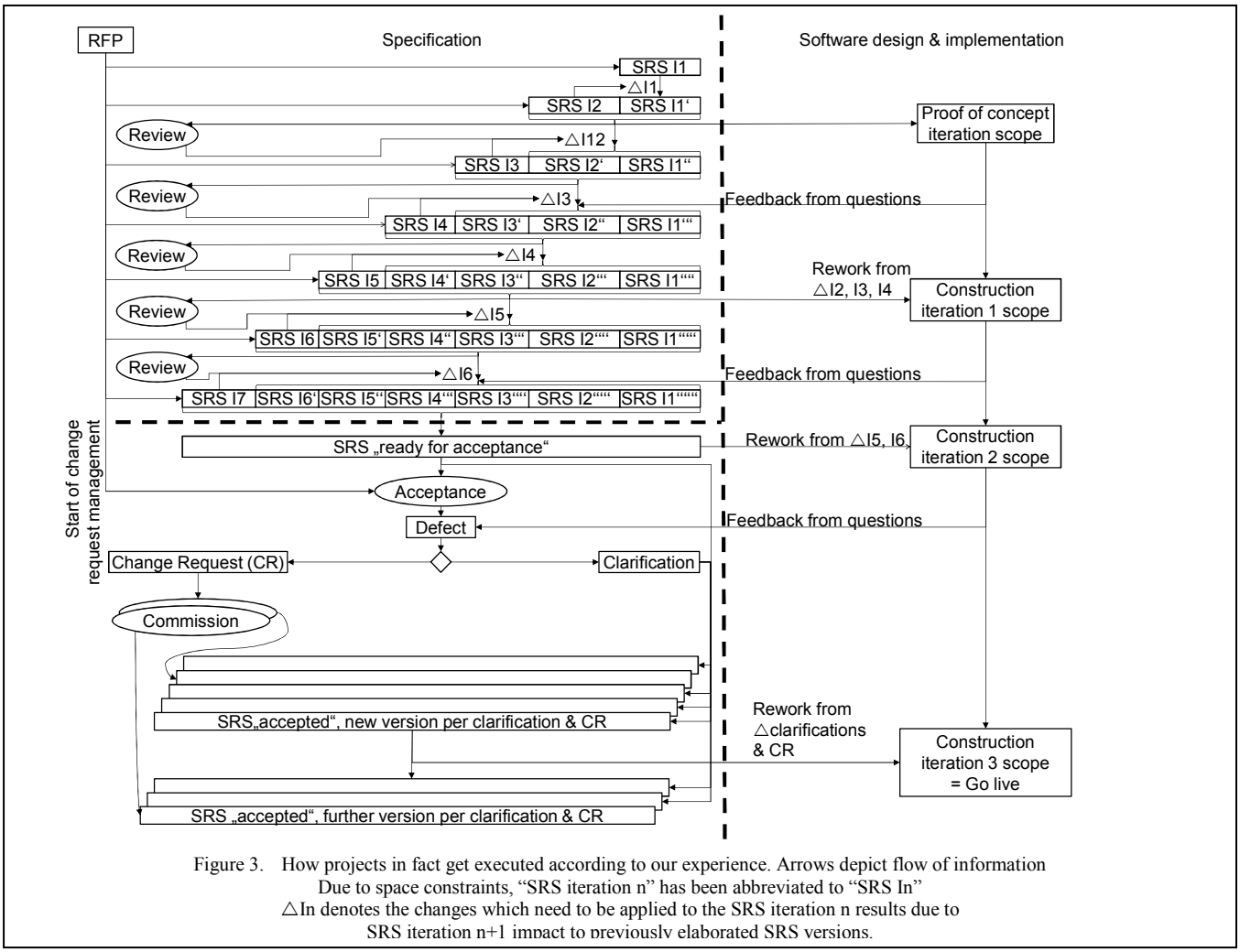

Figure 3. How projects in fact get executed according to our experience. Arrows depict flow of information
Due to space constraints, “SRS iteration n” has been abbreviated to “SRS In”
△In denotes the changes which need to be applied to the SRS iteration n results due to SRS iteration n+1 impact to previously elaborated SRS versions.

“avoidable creep” in the sense of [1]. Finally, the detailed SRS for such systems may end up with a total of some 2000 to 5000 printed pages, typically including requirement specifications, rule configuration specifications, supplementary functional specifications, interface specifications, end user dialogue requirements including screen mockups (user interface design) with lots of forms per screen and screen navigation (one per end user dialogue specification).

As a side effect, the early construction iteration cannot be started with major parts of the functionality being specified. Only about 40% of the functionality could be covered by these iterations. The later construction iterations’ scope needs to be extended considerably.

2) Due to interrelationship between SRS artifacts elaborated in early SRS iterations and those elaborated in later SRS iterations, SRS extension and rework becomes necessary in SRS artifacts elaborated in early SRS iterations. Some SRS artifacts even need an update in each SRS iteration. In the example above, these would be for instance customer initial registration requirements and customer master data maintenance requirements. Such requirements rework typically is not planned in advance. We look at such requirements rework as “avoidable creep” in the sense of [1] as well.

As a side effect, major implementation rework becomes necessary for functionality that has already been implemented during early construction iterations in a pre-final-like way. In that case, from our perspective pure agile approaches do not help since they are still more far away from fixed price settings than our suggested solution is. We observe that up to 25% of the functionality initially implemented during the first and second construction iterations need rework.

3) After the customer accepted the SRS as RFP requirements specification section replacement, change request management becomes the most important task to manage the project scope. For several reasons (pure typographical corrections as well as “avoidable creep” and „unavoidable creep“), up to 10% of the project scope is subject to change after the SRS acceptance. „Avoidable creep“still occurred after the RFP requirements replacement. But due to solid business analysis (requirement engineering), the „avoidable creep“ considerably decreases in terms of effort when compared to what subcontractors have to spend on managing creep before the RFP requirements

replacement. Along with the change request management, disciplined release management has to be introduced early enough to keep implementation controllable in terms of scope.

Fig. 3 illustrates how such projects in fact get executed.

III. PROBLEM STATEMENT

The remainder of this paper is to describe our suggestion to answer the question

> How to organize a project to avoid "avoidable requirements creep" and thereby to reduce subsequent implementation rework?

In other words: We will suggest a procedure of how to make management really perceive the impact of scope creep.

IV. THE SUGGESTED SOLUTION

The basic idea is to spend money in a way that enforces communication about scope and process as soon as possible. In the words of [1] this means to valuate with cost not only "unavoidable creep" but also „avoidable creep".

To get this idea to work, two steps are needed:

1. An SRS *acceptance* step after each and every SRS iteration. The RFP requirements specification section is replaced by the SRS to the extent to which the SRS has been elaborated by the end of each SRS iteration. The plan of payments (Fig. 5) reflects this by having payments planned for each SRS iteration instead of one payment for the final SRS version.
2. To start *change* request management as soon as the first SRS iteration has been finished, which means right after the first SRS iteration. Every change to a requirement that was specified in the SRS iteration n-1 should be viewed at as a change request in the SRS iteration n. This means to apply change request management not only to the implementation but to the SRS itself as well. And it means to apply change request management after each and every iteration and phase regardless of which artifacts were to elaborate during that iteration or phase.

We believe that the combination of *stability* achieved by acceptance steps for even parts of the SRS and *flexibility* achieved by change request management for accepted parts of the SRS motivates all parties to spend money in precisely the way that necessary negotiations are initiated as early in the project as possible.

A. The real scope of construction iteration n (Fig. 4)

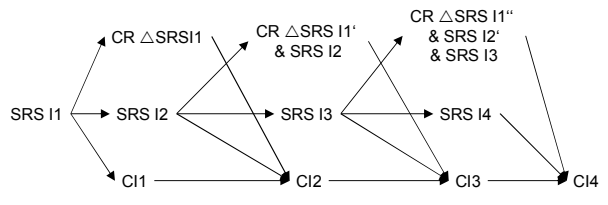

Figure 4. Construction iteration n scope. Due to space constraints, "SRS iteration n" has been abbreviated to "SRS In" and "Construction iteration n" has been abbreviated to "CIn"

During the elaboration of the second SRS iteration, the first SRS iteration version (meanwhile accepted as RFP requirements specification section replacement) might need to be changed due to interdependencies between the first and the second SRS iteration scope. Each of these changes should be viewed as a change request even if the IT system scope has not yet been specified completely. But the scope specified during the first SRS iteration should be viewed as being completely specified because it has been accepted as RFP requirements specification section replacement to the extent by which the first SRS iteration covers the scope. And each of these change requests should be negotiated and decided prior to the second SRS iteration RFP requirements specification section replacement acceptance.

The real, the full scope of the second construction iteration then is described by the second SRS iteration result plus scope specified by change requests to the first SRS iteration scope. More generally, the construction iteration n scope is equal to the SRS iteration n result plus accepted change requests to SRS iteration n-1 scope. Once the SRS iteration n scope together with the change requests to SRS iteration n-1 have been accepted, the SRS iteration n-1 result should be updated according to the change requests to document the scope coherently instead of a documentation of scope consisting of a huge amount of documents just describing delta to some other documents.

Reading and understanding such a fragmented SRS will take a huge amount of time whereas documents being updated according to change requests will considerably improve the reader's ability to understand. The same approach to change requests and documentation updates is valid during IT system maintenance.

B. The plan of payments (Fig. 5)

A figure of 5.00% in cell "SRS I2/SRS delivery" means: Upon delivery of the second SRS iteration results, the customer is obligated to pay 5.00% of the fixed price amount. A figure of 3.33% in cell "CI1/User acceptance test passed" means: Upon the first construction iteration having passed the user acceptance test, the customer is obligated to pay 3.33% of the fixed price amount.

The cell figures are calculated to be the product of

- by which percentage the *scope* shall be increased by the iteration in focus
- times the percentage to be paid upon the *trigger* in focus
- times the percentage of *effort* spent in requirements and implementation, respectively.

Change requests will apply only to those parts of the SRS (= the amount of scope) that have been accepted as RFP requirements replacement in advance.

C. The flow of payments

At the very beginning of the project, right upon signing the fixed price contract, the supplier may bill 20% of the price. Assuming that the IT system could be deployed into production as a whole only, another 20% of the price (30% of 2/3 of the whole price) could be billed at the very end of the project. Between these two payments, another 20

payments are planned quite equally distributed over time. The first and second payments are those callable upon the first SRS iteration results delivery and the acceptance of the RFP requirements specification section replacement for the first SRS iteration, followed by second SRS iteration, first construction iteration, third SRS iteration, second construction iteration etc. payments timely interwoven depending on when the payment-triggering events occur.

		Scope planned to be increased by x%		20%	30%	30%	20%
		Total of scope achieved at the iteration end		20%	50%	80%	100%
Effort spent for	Payment trigger		Amount				
Requirements (1/3)			33%	SRS I1	SRS I2	SRS I3	SRS I4
	Contract signment		20%	1.33%	2.00%	2.00%	1.33%
	SRS delivery		50%	3.33%	5.00%	5.00%	3.33%
	RFP req. section replacement acceptance		30%	2.00%	3.00%	3.00%	2.00%
Implementation (2/3)			67%	CI1	CI2	CI3	CI4
	Contract signment		20%	2.67%	4.00%	4.00%	2.67%
	Implementation delivery		25%	3.33%	5.00%	5.00%	3.33%
	User acceptance test passed		25%	3.33%	5.00%	5.00%	3.33%
	IT system productive deployment		30%	4.00%	6.00%	6.00%	4.00%
		Change requests might apply to a maximum of x% of scope		0%	20%	50%	80%

Figure 5. Plan of payments when implementing the suggested solution. Due to space constraints, "SRS iteration n" has been abbreviated to "SRS In"and "Construction iteration n" has been abbreviated to "CIn"

The initially planned flow of payments may be extended by payments for change requests as soon as the first SRS iteration has been accepted as RFP requirements specification section replacement.

D. How to plan SRS iteration scope

To assign SRS artifact elaboration to SRS iterations, a valued and prioritized list of SRS artifacts is necessary at the very beginning of the project.

The first step to arrive at this list is to do a rough result-oriented breakdown of the SRS. For instance, business process analysis could serve to arrive at a list of requirements. A list of supplementary functional specifications and other SRS artifacts should be elaborated as well based on a business and IT system architecture draft.

The second step is to prioritize the SRS artifacts' elaboration. When prioritizing SRS artifacts' elaboration, not only technical dependencies should be considered but also project communication aspects. By project communication aspects we mean for instance to prioritize elaboration of artifacts which could serve for marketing the project progress to the customer. In some of the projects reported about here, this meant to specify a subset of requirements by which the general contractor could show to the customer a walk through the whole set of business processes.

The third step is to value each SRS artifact by a three-value complexity attribute (high, medium, and low complexity). The sum of these is the complexity value of the whole project.

The fourth step is to assign SRS artifact elaboration to SRS iterations according to the prioritization order such that each SRS iteration set of SRS artifacts summed up complexity value equals to the respective percentage of the overall sum of complexity values.

The SRS iteration assignment of SRS artifact elaboration should be agreed between all parties and become part of the fixed price contract. If during an SRS iteration either the list of SRS artifacts or their prioritization or complexity value needs to be changed, then such change must result in a change request even in very early stages of the project.

V. BENEFITS TO THE CUSTOMER

Fig. 6 shows the benefits to the customer together with explanations about possible challenges in achieving the benefits and ways to cope with these challenges.

VI. SUGGESTIONS FOR FUTURE RESEARCH

Especially in large projects, there is a tension between the top management's need to have a detailed SRS at the very beginning of the project to provide a valid fixed price effort estimation, and the fact that project scope cannot be stated in that detail at the very beginning of a project as a matter of principle. We strongly believe that this dilemma needs more consideration and needs to be backed by facts to enable the right communication at the right point in time. The following questions surround the one which we suggested a solution for in this paper:

1. How to *still providing* to the customer the comfort of a fixed price environment while at the same time enforcing scope and budget discussion much earlier in the project according to the procedures suggested in this paper?
2. How to *estimate* and consider in the fixed price the size of implementation rework due to "avoidable creep"?
3. How to *design* the software such that implementation rework is minimized in each phase?

The challenge is to get these questions answered based upon an RFP since this is the only information available at the point in time where these questions need to get answered.

As far as the first question is concerned, we anticipate that customers will not perceive the solution suggested in this paper as a pure fixed price contract. As we pointed out in section I and Fig. 6, thorough communication might help to cope with this problem and get the customer to understand and valuate positively the risk reducing effects of the suggested solution. But by now, even by thorough communication some customers might not be willing to understand this aspect.

For this reason we still find the first question not to be answered completely. The second and third question we have not been able to find answers yet. Maybe all three of them could become subject to exchange between practitioners or even future research.

VII. CONCLUSION

The question we raised in the problem statement section III (How to avoid requirements rework due to "avoidable creep") has been answered sufficiently from our perspective. When a supplier makes use of our suggested solution, the

Benefit	Possible challenges
Heading for a true win-win-situation: Customer and supplier proceed jointly on safe ground	If the supplier claims change requests for every change at the very early stage of the project as we suggest it in this paper, the supplier will risk being called a nitpicker, killjoy and spoilsport. As everybody does who comes up with the term "change request" and other formalisms at the very beginning of a project when everybody wants to work on the content but not on the process. However, if the background would be explained to the customer e.g. by referring to the experience described in this paper, we strongly believe that by thorough communication the risk of being misunderstood could be reduced considerably and maybe even replaced by a sound understanding of the customer's benefit: Establishing a solid ground for creating the new system.
Reducing tension (delay) at the end of the project (acceptance): Necessary negotiations will take place as early as possible	The customer would need to take over responsibility for the SRS much earlier than in a pure waterfall-like scenario. The customer however might not be willing to take over that responsibility at the agreed point in time due to not clarified issues like "How to negotiate and prove that an SRS iteration result covers the respective part of the RFP requirements specification section?" or "Is some specific functionality a change request or not?" In projects like the ones described above, such negotiations usually take place while the team elaborates the SRS-to-RFP coverage mapping. This mapping usually is elaborated subsequently to the SRS elaboration. This point in time however usually is too late to avoid requirements and implementation rework caused by "avoidable creep". We think the transfer of responsibility will raise exactly those questions as early as possible.
Initiating negotiations as early as possible: Flow of payments supports to initiate negotiations.	Selling the project might need lots of explanation in advance since the plan of payments described in Fig. 5 might appear quite complex and not self-explaining at all. However, we think that by thorough communication it could be made understood that the flow of payments reflects the transfer of responsibility for the specified functionality. Assigning cost to the transfer of responsibility is assumed to be the most important trigger to raise as soon as possible the negotiations described above.
Improving self-motivated coordination: Flow of payments motivates the supplier to work coordinated	According to the suggested solution, the payments flow more continuously than in a pure waterfall-like scenario but still in a well-controlled and discrete manner. *It is still not "time and material"!* The supplier does not depend that heavily upon foreign liquidity like bank loans to pay his employees' salaries as it is the case in a scenario where most of the fixed price will be paid not before delivery or even acceptance of the system as a whole. This will motivate the supplier to work more coordinated with the customer instead of implementing and billing something which does not really meet the customer's expectations.
Building mutual confidence to build the right system	It might be challenging to plan phases and iterations in a way that core and self-contained functionality will be implemented first. But we believe that planning the project in this manner creates the confidence that in the end a system will be created that satisfies solidly what the customer wants.
Processsing uniformly any type of requirements creep	Sometimes it is not clear whether some requirements change is "avoidable" or "unavoidable". Maybe RE effort was cut because nobody knew about some specific functionality earlier than the respective change request appeared. Such may root e.g. in processes of learning about intended scope and functionality. Such processes happen in each and every project. The suggested solution considers both types of creep uniformly, in a disciplined way, and on the detailed single SRS requirement level.

Figure 6. Benefits to the customer when applying the suggested solution, together with possible challenges in achieving the benefit, and ways to cope with these challenges

customer's management will be tackled by change request and scope negotiations as soon as possible. The effort to get a change request accepted is much higher than just changing an SRS artifact. This difference in effort we anticipate to enforce two things at a very early point in time: The "Do we really need that gold plating" discussions and the need to focus on the most complex aspects first since every change at a later stage will result in additional cost.

However, as we pointed out in section VI, our coherent set of questions remains not fully answered and may become subject to research in the future.

Finally, we would like to emphasize that, from our perspective, the suggested solution serves as a step in converging pure waterfall and pure agile procedures.

REFERENCES

[1] D. M. Berry, K. Czarnecki, M. Antkiewicz, M. AbdElRazik, "Requirements Determination is Unstoppable: An Experience Report," re, pp.311-316, 2010 18th IEEE International Requirements Engineering Conference, 2010.

[2] A. Cockburn, Multiple web pages having been accessible on June 6th, 2011 at http://alistair.cockburn.us/Crystal

[3] N. Hanakawa, K. Okura, "A Project Management Support Tool using Communication for Agile Software Development," apsec, pp.316-323, 11th Asia-Pacific Software Engineering Conference (APSEC'04), 2004

[4] J. Lewis, K. Neher, "Over the Waterfall in a Barrel - MSIT Adventures in Scrum," agile, pp.389-394, AGILE 2007 (AGILE 2007), 2007