Supporting Effective Collaborative Engineering[1]

Joerg M. Haake & Till Schümmer

Department of Mathematics and Computer Science
FernUniversität Hagen
Universitätsstr. 1
58084 Hagen, Germany
{joerg.haake|till.schuemmer}@fernuni-hagen.de

Abstract: Facilitating and managing the development and application of effective shared work practice in distributed teams remain a key challenge for effective collaborative engineering. We propose integrated support for project management, collaboration and engineering activities through dedicated task patterns, which are combined into a holistic model of collaborative engineering practice. Flexible enactment support for this model facilitates system and human enactment while using collaboration services as a front-end to project management and engineering facilitates effective teamwork. Finally, a reflexive learning cycle fosters the development of improved shared practice. Experiences in three industrial pilots suggest the applicability of the approach.

1 Introduction

Due to globalization design and manufacturing are increasingly performed by distributed networked organizations. Rapid changes of the economic environment demand faster responses and techniques for implementing agile organizations become more important. These agile organizations are characterized by distributed teams consisting of team members from different partner organizations working collaboratively on a joint project (e.g. collaborative product design or manufacturing).

Current support for such distributed teams includes different groupware technologies facilitating to some extent sharing of artifacts, communication, and coordination. Project or team portals provide unified access to shared artifacts and otherwise isolated tools. However, the difficult problem of facilitating and managing the development and application of effective shared work practice in distributed teams remain as of yet unsolved.

In this paper, an approach is described that addresses this problem. Through the integration of project management, engineering, and support for shared work practice more effective collaborative engineering becomes possible.

[1] This work was supported by the CEC under grant FP6-IST-016527 MAPPER

The remainder of the paper is organized as follows: Section 2 analyzes the problem and reviews the state of the art. Section 3 then introduces the approach followed by a description of its prototypical implementation in the FP 6 MAPPER project (section 4). Section 5 presents initial experiences while section 6 summarizes the main results, compares them to prior research, and concludes with open research questions.

2 Problem Analysis and State of the Art

Collaborative engineering denotes the joint practice of creating an engineering solution by a team. Organizations performing collaborative engineering can be regarded as a socio-technical system. Effective collaborative engineering requires both, guidance and best practice for team members when interacting in the team and matching technical mediation of these interaction processes. Furthermore, since engineering and design are regarded as ill-structured problems [Het02, RW73] agile support matching the ever changing needs of the team members is needed.

In industry, such collaborative engineering takes place in the context of a business process and is usually organized and managed as a project. Figure 1 shows the relationships between these three aspects: Project management is concerned with project planning and project execution in the context of the larger business (e.g. its objectives, resources and constraints). Project execution includes resource allocation, monitoring progress and risks, and adapting the project plan according to changes in the environment. Collaboration includes communication among team members, coordination of their activities, and cooperation of team members to reach the team's goals. While project management steers a project on a coarse level, collaboration organizes the actual work practices shared by the team. Only if these shared work practices match the requirements of the project management and of the engineering domain, effective collaborative engineering can take place. Finally, engineering includes domain specific constructs and methods of creating a working design as well as the design constraints that must be met.

Engineering can be seen from two perspectives: Firstly, the product centered perspective of engineering focuses on the representation of the design (e.g. to be used for manufacturing) and on the respective design constraints (which may be explicitly represented and checked by, e.g., a model checker). Secondly, the task centered perspective of engineering focuses on a (rough) planning of engineering tasks and respective milestones. Here, engineering methods are often left implicit or put into check lists while their application remains entirely with the engineers.

From figure 1 it becomes apparent that effective collaborative engineering (i.e. a practice that minimizes friction in the team while maximizing design quality in terms of matching the requirements and development speed) requires an approach integrating the above three aspects.

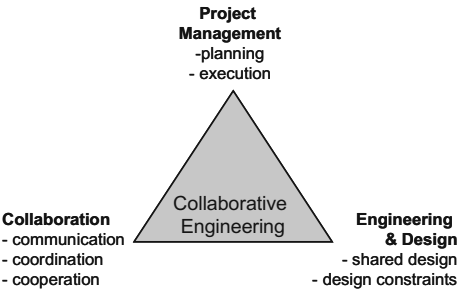

Figure 1: Aspects of socio-technical systems in collaborative engineering

Collaborative engineering deals with multiple engineers working jointly on a shared complex design, synchronously and asynchronously. Efficient collaborative engineering in larger projects requires integrated support for the above three aspects. Since these aspects are intertwined, it is not sufficient to provide separate support. In complex design projects, engineers cannot be expected to manage these interdependencies efficiently without explicit tool support.

In the following we discuss current approaches for integrating different aspects of collaborative engineering (cf. figure 1). Current approaches support the integration of up to 2 of the 3 aspects:

Using collaboration support in engineering: A plethora of groupware tools support different aspects of collaboration, which can also be used to support asynchronous and synchronous collaboration during design. IM tools (e.g. MSN or Windows Life Messenger, cf. http://get.live.com/messenger/overview) and A/V conference tools (e.g., Skype (http://www.skype.com/), FlashMeeting (http://flashmeeting.open.ac.uk/) support informal communication, while application sharing systems such as VNC (http://www.realvnc.com/) allow sharing design applications. However, application sharing of single user design applications provides only limited collaboration awareness, which may lead to coordination problems. Shared workspaces (e.g. BSCW [Klö02]), Wiki's [LC01], and document repositories can be used to share and, to some extent, jointly edit documents. Finally, collaboration platforms including support for sharing and joint manipulation of artifacts as well as coordination and communication support (e.g. CURE [HS+03], Lotus Notes (http://www-142.ibm.com/software/sw-lotus/products/product4.nsf/wdocs/noteshomepage)), and collaboration portals providing access to data sources, shared tasks, shared artifacts and otherwise isolated tools (such as Metis Information Portal (http://www.troux.com/products/metis_server/), SAP Netweaver Portal (http://www.sap.com/platform/netweaver/)) provide integrated access to tools and services required within collaborative task execution. However, these generic platforms and portals do not provide any integration of project plans with engineering specific processes and tools.

Integrating project management with engineering: ProjectCards [AO+06] is an example of an Integrated Product Development Environments (IDE) with project management support. It is a commercial plugin for Eclipse (http://www.eclipse.org/), which supports real time sharing of XP project plans resp. XP stories [Bec99]. However, ProjectCards supports only limited communication among developers, limited

integration of plan and software versions, and decision making is not supported. Other examples are engineering environments with limited collaboration capabilities. CAD tools, e.g. CATIA (http://www.3ds.com/products-solutions/plm-solutions/catia/), provide repository access, versioning, task support and communication tools but fail to facilitate and manage the development and application of effective shared work practice in distributed teams.

Integrating collaboration and engineering: cooperative design editors allow multiple engineers to discuss, share and jointly manipulate designs, while application sharing systems such as VNC (http:// www.realvnc.com) allow the sharing of entire single user design applications. However, application sharing of single user design applications provides only limited collaboration awareness, which may lead to coordination problems as coordination and the development of shared work practice is not supported (cf. the SHARED EDITOR pattern and the APPLICATION SHARING pattern in [SL07]).

Integrating project management and collaboration: Here, project planning and management tools (e.g. MS Project (http://office.microsoft.com/en-us/project/)) and workflow engines for coordinating task assignment and execution (e.g. work list handlers such as WorkWare [JC99]) have been developed. These approaches focus on supporting standardized work flows. However, design processes are often very creative and require flexible emergent combination of steps to tackle ill-structured problems. So far, IDEs provide only limited support for combining project management and collaboration in a flexible manner. TMRS (http://ecolleg.org/trms/) is an example for combining definition of design workflows with secure tool invocation resulting in a simplified execution of design processes requiring different tools from different partners. However, collaboration among team members is not its focus.

In summary, none of the above approaches support the needed integration of project management (i.e. support for high level planning and execution), collaboration (i.e. support for development and application of required shared work practice) and engineering. In addition, the above approaches are focusing on the technical part of the socio-technical system of collaborative engineering. Thus, they do not support the social practice of collaborative engineering very well. The next section introduces our approach to these problems.

3 Approach

The goal of our approach is the integration of project management, collaboration, and engineering in order to support facilitation and management of the development and application of effective shared work practice in distributed teams. We distinguish two dimensions of supporting work practices in a socio-technical system: tailorability and enactment. Tailorability is concerned with the ability of team members to adapt work practices to their needs while enactment distinguishes enactment of work practices by a technical system (e.g. workflow management system) or by team members. These dimensions span a design space for socio-technical systems. Non tailorable work practices lead to system-enacted strict workflow solutions and human-enacted strict application of methods or algorithmic procedures. These approaches are not applicable

to collaborative design. However, tailorable work practice as supported in a semi-automatically enacted way in flexible workflow systems or through team members following a dedicated interaction pattern address the needs of collaborative engineering for agility and user guidance.

We propose to use interaction patterns for supporting human enactment of project planning, collaboration, and engineering work practice. In addition, we propose to use visual models for supporting automatic enactment of modeled work practice. Patterns [Ale79] capture solutions to problems at a more abstract and implementation independent level. They are used to guide users in the problem solving activity in their concrete environment. Models [Sch06] capture properties and relationships of entities involved in the respective domain and may provide views and tools for exploring and manipulating the model. If models contain descriptions of procedures they may also provide execution support (and thereby become active knowledge models [LK02]).

Agile organizations require a continuum for facilitating enactment of work practices through patterns and models. While patterns provide more flexibility and adaptability in frequently changing situations, models may provide more efficient execution support in recurring situations. In order to balance the needs of agility and efficiency in an optimal way we propose to combine models and patterns in a flexible way: models are used to express automation aspects while patterns are used to guide and educate team members.

Our approach can be characterized by:

1. Project management, collaboration, and engineering are characterized by models and patterns in a unified model format: each model consists of both descriptive and enactment capabilities through views and tools while the matching pattern descriptions are represented as informal textual model elements and appropriately linked to other model elements. In this way, the model consists of executable process description where possible and more flexible pattern descriptions otherwise. Together, these elements enable the models to provide dedicated problem solving help in a flexible manner. Concrete models will always contain some aspects of patterns and vice versa. In the case where the model is extended with a problem-solution sketch of a pattern, we consider the model as the primary chunk of process knowledge. Such an approach is preferred when the automatic execution of the supported task seems feasible. In cases where such an automatic execution is unlikely because of a too variant context, we consider the textual pattern as primary chunk of process knowledge. The textual pattern is then augmented with a model-based interaction guideline for the team members. Automatic execution of these guidelines is, however, not intended. An example of a pattern collection for one aspect, namely project management, can be found in [ST08] that collected patterns for technology-enhanced meetings. These patterns show how steps of a solution are connected to model-configured services and tools.

2. The models and patterns of the proposed three aspects of collaborative engineering are combined (see Figure 2): The project management model and

patterns guide collaboration and engineering activities through provision of tasks and staff working on them in appropriate roles. The collaboration model and patterns facilitate effective collaboration activities (shared work practice) on shared design, which is facilitated by the product / design model and patterns. The design model needs to acknowledge constraints provided by the project management, and serves as shared objects / artifacts of the collaboration patterns.

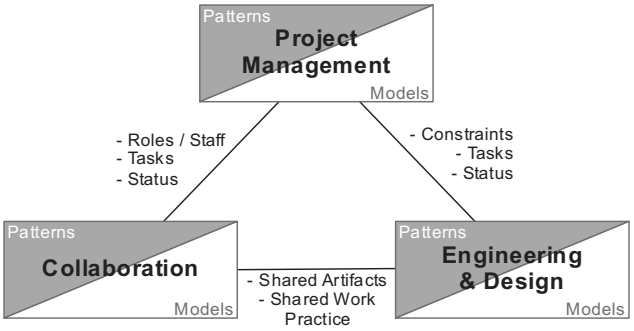

Figure 2: Models and Patterns for Collaborative Engineering

Note that we do not distinguish between patterns and models when considering the links in Figure 2. Instead we propose to use both models and patterns and put the main attention on one of them depending on the concrete problem addressed by the model or the pattern. Together, the three interlinked models form a holistic model of collaborative engineering work practice of the organization. In the holistic model, relationships between patterns form a pattern language for collaborative engineering. Links and reuse of entities between models represent dependencies between the aspects. In our approach we propose to provide patterns tailored to the collaborative engineering environment, thus enabling the provision of some execution support for the problem solving activity in related executable models (such as facilitating the application of a pattern by providing guidance and needed resources in a portal or as parts of an IDE).

3. The enactment of the holistic model at run-time leads to two problems: Firstly, the model must be instantiated after its components have been selected from a template model. Secondly, models or patterns from one aspect (e.g. facilitating collaborative authoring) may be used to facilitate another activity captured in another model or pattern (e.g. manipulating a shared design). Enactment requires here that the collaborative authoring model is injected into the current instance of a shared design model. This injection requires unification and mapping of elements of the two models (e.g. the read and write activities of authoring must be unified with manipulation activities on the shared design).

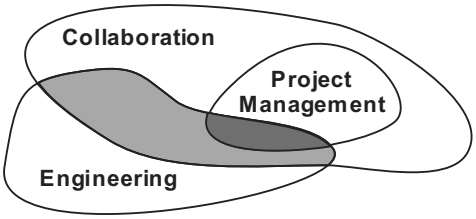

Figure 3: Overlap between the different perspectives on collaborative engineering

4. Since collaborative engineering is by nature a collaborative activity we propose to use collaboration models and patterns as a starting point for project management and engineering activities and product respectively project data. This acknowledges the fact that both, project management and engineering, are subject to collaboration. Thus, team members can collaborate on the engineering as well as on the project management activities. The proposed holistic model defines information flow and dependencies between the respective aspects so that status information from collaboration and engineering are available in project management while resource allocation and task planning in project management lead to respective adaptations of the collaboration and engineering models, which are then reflected in the user interface of the collaborative engineering environment. However, users are still free to act within the information and activity space defined by the respective models.

5. Finally, sharing of models and patterns is an important aspect on an organizational level as well as on a project level. We envision a knowledge sharing and learning process as depicted in Figure 4. This approach is based on the model for creative innovation that was initially proposed in [Shn02] and refined in the FEED FACTS pattern in [SL07b]. A project collects patterns and models from the organization-wide (or inter-organizational) repository (Figure 4-1). The patterns and models are related to each others and set in context with the concrete project (Figure 4-2). During execution of the models and patterns, the project creates a contextualized version of the pattern. In addition, new best practices emerge by adapting already known practices to new challenges (Figure 4-3). The practices need to become explicit which means that the team members need to be supported in a reflection process that forms a learning loop. Here we distinguish two cases: Firstly, in a given project the team facing a break down [Sch83] may use this opportunity to reflect about their way of working, which may lead to changes of their shared work practice reflected in improved models and / or patterns. The improved models / patterns will then be used in the same project and performance is improved [Fer05]. Secondly, an organization may decide to evaluate projects upon completion in a reflection phase in order to identify best practice. Project retrospectives [Ker01] are one way for achieving this. Another approach is the technique of pattern mining, as it is widely practiced in the Design Patterns community. All reflection activities lead to improved practices (Figure 4-4). The mined best practice should then again be encoded in template models and patterns, which may be donated to

other projects and form the basis for setting up similar projects in the future (Figure 4-5). This will lead to improved performance in future projects. Here, models and patterns of the concrete projects serve as a data base for reflection about past experience.

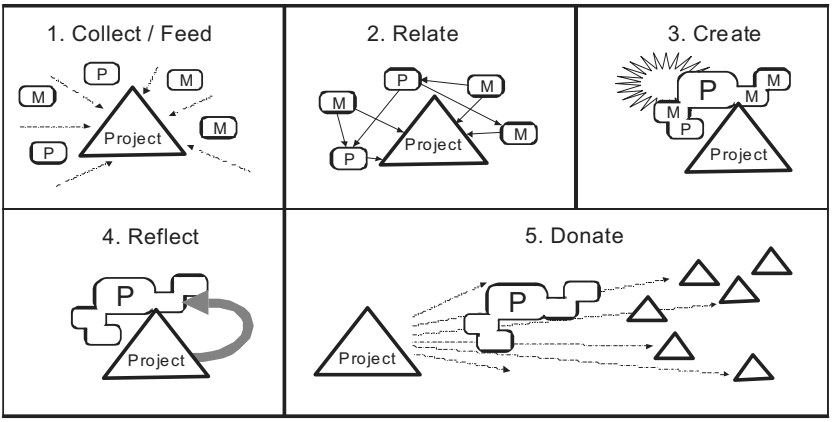

Figure 4: Knowledge sharing and learning in collaborative engineering (P=patterns, M=models)

4 Implementation

First parts of the proposed approach have been implemented in the MAPPER project. MAPPER is concerned with supporting networked manufacturing through adaptive model-based process and product engineering. From a model perspective, the Active Knowledge Model (AKM) technology [LK02] is central to MAPPER. An Active Knowledge Model is a visual externalization of knowledge of enterprise aspects that can be operated on (viewed, traversed, analysed, simulated, adapted and executed) by industrial users [Lil99]. The visual model must be available to the users of the resulting information system at runtime. Furthermore, the model must influence the behaviour of the computerised support system. Finally, the model must be dynamic, i.e. users must be supported in changing the model to fit their local needs, enabling tailoring of the system's behaviour.

While creating models that capture best practices, we identified *task patterns* as a combination of visual models with a textual representation of three aspects, namely a context description, a problem statement, and a description of the solution (note that this triple is present in most patterns found in the design patterns literature). The task patterns bring together sequences of engineer's activities, design workflows or best practices [PP+06] (p.14). It became clear that model execution as it was proposed by the AKM approach has its limitations when facing very agile design contexts. We thus decided to complement the strict task pattern models with descriptions of work practices that are intended to be executed by the concrete team member in a design team.

Our current technology supports the user in creating visual models for task patterns that can be linked to tools for collaboration. The models help to configure the respective collaboration support tools within the current workflow of design activities. They are further used to generate a web-based portal that serves as a control center and orchestrates the different tools. To start a model-based micro-workflow, the user instantiates a visual model of the task pattern and the enactment engine creates the appropriate tasks in the portal and configures the required tool.

Complementary to this system-automated enactment of models and task patterns, we have tested means for human centered task enactment. In these cases, the users play the role of the service orchestrator and translate the different task patterns into their concrete design context.

The learning loop introduced as the last step of our approach is currently supported by allowing team members to alter and appropriate models as well as patterns. In workshops among users and workshops within a methodology team, these new emerging solutions are discussed and generalized so that they can be used in a broader context. The technical support for such activities is, however, still in its infancy.

5 Experiences

Our approach is currently applied in three pilots within the MAPPER project. Each pilot focuses on a different application of collaborative engineering: pilot 1 deals with collaboration between a car manufacturer and its suppliers during the design phase, pilot 2 deals with collaborative engineering of complex chips, and pilot 3 deals with collaboration between a automobile part manufacturer and its suppliers during innovative design projects.

In all three pilots, task patterns and models were created, although with a different focus and extent in each pilot. While pilot 1 emphasized the shared design model and its collaborative use, pilot 3 focused on a shared product model and supporting its use and management through respective task patterns and models. Pilot 2 emphasized support for shared access to a joint development environment respective product data and collaboration patterns facilitating distributed design and bug tracking.

Initially, task patterns and models were created in workshops with users. Over time, users took more ownership and started modeling and tailoring themselves. Although the first field tests are still ongoing, early observations indicate that users are capable (with some help) to tailor their models used in their projects and that they can map task patterns to their context. Not surprisingly, the provision of explicitly shared task structure seems a helpful guidance for developing shared practice. We expect more detailed feedback from the running formative evaluation studies.

6 Conclusions

In this paper, we presented an approach for facilitating and managing the development and application of effective shared work practice in distributed teams. Our approach supports project management, collaboration and engineering / design through

- dedicated task patterns for each aspect of collaborative engineering,
- forming a holistic model of collaborative engineering practice through combining the above task patterns,
- flexible enactment support that distinguishes between system and human enactment,
- using collaboration as a front-end to project management and engineering, and
- a reflexive learning cycle leading to improved shared practice.

Our approach exceeds prior research in multiple ways: While previous approaches neglected the integration of project management, engineering, collaboration and learning, this approach aims on integration. Furthermore, using task patterns in an integrated way, socio-technical systems can be designed in a holistic way by the users themselves. Representing both project data and best practice templates as task patterns facilitates reflective learning and the development of shared practice.

Although it is too early to provide final answers, our first experiences with the approach in three pilots indicate its principal applicability.

One problem that we already identified was that models may be too restrictive when it comes to the point of instantiating the task pattern. The challenge lies in injecting other task patterns at this time. We are faced with technical issues as well as complexity problems here. When we want to support the users in creating a homomorphism between different components of a model, we have to allow the user to link from within a task pattern to components of other task patterns. The benefit from hiding internals of a task pattern from the user is lost in such a case and the process of wiring the different components showed to be a challenging task for all stakeholders. Patterns may be more flexible here, however, their flexibility results from the fact that the homomorphism is not made explicit, which again can cause problems in understanding the current work practices during a breakdown situation. Finding a better and user friendly approach to this problem will be an important challenge for future work.

References

[Ale79] Alexander, Christopher. *The timeless way of building*. Oxford University Press, 1979.

[AO06] Astles, Graham; Ouellet, Benoit; Theriault, Jeff. ProjectCards. (Online: http://www.projectcards.com) 2006.

[Bec99] Beck, Kent. eXtreme Programming Explained. Addison Wesley, 1999

[Fer05] Fernandez, A.: *Groupware for collaborative Tailoring*, Ph.D., University of Hagen, 06/07/2005.

[Haa02] Haake, J. M. 2002. Cooperative learning and execution of work processes. *IJCEELL* 12(5/6). Inderscience Publishers.

[Het02] Marike Hettinga, Understanding evolutionary use of groupware, Telematica Instituut Fundamental Research Series, Universal Press, Veenedaal, The Netherlands, Enschede, The Netherlands, 2002.

[HL03] Haake, J. M., Lillehagen, F. Supporting Evolving Project-Based Networked Organisations. CE: The Visison for the Future Generation in Research and Applications, 2003, pp. 735-742.

[HS+03] Haake, J. M.; Schümmer, T.; Haake, A.; Bourimi, M. & Landgraf, B.: *Two-level tailoring support for CSCL*. In: Groupware: Design, Implementation, and Use. Proceedings of the 9th International Workshop (CRIWG 2003), Springer, 2003, 74-82.

[JC99] Jørgensen, H. D. and Carlsen, S. Emergent Workflow: Integrated Planning and Performance of Process Instances, Workflow Management '99, Münster, Germany, 1999.

[Ker01] Kerth, N. L.: *Project Retrospectives: A Handbook for Team Reviews*. Dorset House Publishing, 2001.

[Klö02] Klöckner, K. *BSCW. Cooperation support for distributed workgroups.* In: 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing., IEEE Computer Society, 2002, 277-282.

[LC01] Leuf, B. & Cunningham, W. *The Wiki Way*. Addison Wesley, 2001.

[Lil99] Lillehagen, F. Visual Extended Enterprise Engineering Embedding Knowledge Management, Systems Engineering and Work Execution, IEMC '99, IFIP International Enterprise Modelling Conference, Verdal, Norway, 1999.

[LK02] Lillehagen, F. M., Krogstie, J. 2002. Active Knowledge Models and Enterprise Knowledge Management. In Proc. Of ICEIMT'02 - International Conference on Enterprise Integration Modelling Technology held at the University of Valencia, Spain, 02-04-24-26. Kluwer.

[PP+06] Pawlak, A., Penkala, P., Sakowski, W., Wisla, A. & Grau, G. *Deliverable D5: IP-based SoC design requirements model*. MAPPER, Model-based Adaptive Product and Process Engineering, IST/NMP Project, 016527, September 21st 2006.

[RW73] Rittel, H. and Webber, M.; *Dilemmas in a general theory of planning*, Policy Sciences 4, Elsevier Scientific Publishing, Amsterdam, 1973, pp. 155–159.

[Sch83] Schön, D. A. *The Reflective Practictioner: How Professionals Think in Action*. Basic Books, 1983.

[Sch06] Schmidt, D. C., Model-Driven Engineering, *IEEE Computer,* February, 2006, pp. 25-31.

[Shn02] Shneiderman, B.: *Creativity support tools*. Commun. ACM, ACM Press, 2002, 45, 116-120.

[SL07] Schümmer, Till & Lukosch, Stephan. *Patterns for Computer-Mediated Interaction.* John Wiley & Sons, Ltd., 2007.

[SL07b] Schümmer, Till & Lukosch, Stephan. READ.ME – Talking about computer-mediated communication. In Proceedings of EuroPLoP06. UVK, Konstanz, 2007.

[ST08] Schümmer, Till & Tandler, Peter. *Patterns for Technology-Enhanced Meetings.* In Proceedings of EuroPLoP07. UVK, Konstanz, 2008 (accepted for publication).