# Content

Content 1

1. Introduction 3

1 Project Management Basics 4

1.1 Deliverables 4

1.1.1 Requirements Analysis Document (RAD) 4

1.1.2 System Design Document (SDD) 5

1.1.3 Object Design Document (ODD) 5

1.1.4 Code of Prototype 6

1.1.5 Test Manual 6

1.2 Typical Project Elements 6

1.2.1 Kick-off meeting 6

1.2.2 Team Meetings 7

1.2.3 Management meetings 7

1.2.4 Client visits 7

1.2.5 Internal reviews 8

1.2.6 Client Acceptance Test 8

1.3 Management Roles 8

1.3.1 Methodologist 8

1.3.2 Project manager 8

1.3.3 Coach 9

1.3.4 ‘Developer 10

1.3.5 Technical Writer 10

1.3.6 Client 10

2 Meetings 11

2.1 Types of meetings 11

2.1.1 Information only 11

2.1.2 Brainstorming 11

2.1.3 Decisive 11

2.2 Meeting Roles 11

2.2.1 Primary Facilitator 12

2.2.2 Recorder 12

2.2.3 Minute Taker 13

2.2.4 Time Keeper 13

2.3 Sample Meetings 13

2.3.1 Kick-off Meeting 13

2.3.2 First Management Meeting 13

2.3.3 First Team Meeting 14

2.3.4 Meeting with Student Primary Facilitator 15

2.3.5 First Regular Team Meeting 15

2.4 How to make Meetings Work 15

2.4.1 Judging a meeting 15

2.4.2 Group memory 16

2.4.3 Things that can go wrong and how to solve them 16

2.5 How to deal with problem people 18

2.5.1 The Latecomer 18

2.5.2 The Early Leaver 18

2.5.3 The Broken Record 18

2.5.4 The Doubting Thomas 18

2.5.5 The Head shaker 19

2.5.6 The Dropout 19

2.5.7 The Whisperer 19

2.5.8 The Loudmouth 19

2.5.9 The Attacker 19

2.5.10 The Interpreter 20

2.5.11 The Gossiper 20

2.5.12 The Know-It-All 20

2.5.13 The Backseat Driver 20

2.5.14 The Busybody 20

2.5.15 The Interrupter 21

2.5.16 The Teacher's Pet 21

3 References 22

# Introduction

This handbook intents to assist new instructors of upcoming projects to lead their teams to a successful Client Acceptance Test. The nature of our projects focuses on innovation and we either introduce new requirements or new business rules or new technology such as shared databases, wireless. In particular we are also interested in new processes and methodologies. In university courses the students have to be taught how to deal with new requirements, technologies and methodologies. In industry you could use this guide for a pilot project, in which the introduction of a new technology or methodology is investigated.

This handbook tries to help you as instructor to cut the project start-up time to a minimum and help your group to become more productive. When the infrastructure works and the meeting process is clear, a project gains more momentum and meetings become more efficient. Therefore this handbook provides heuristics, advices and guidelines. Documents templates and examples from the companion website aid you as instructor to produce the necessary documents. Chapter 2 of this document introduces important terms and some project management basics. Chapter 3 gives you an overview of the different types of meetings you will attend and what your responsibilities will be within these meetings.

# Project Management Basics

Before looking at hints on how to be a better instructor, you should be familiar with a number of terms. This chapter first discusses the goals of a project - the project deliverables, i.e. the results your 'customer' will 'pay' for. Then you are given an overview of typical elements of a project. The last part of chapter 2 explains key project management roles.

## Deliverables

When a project is over, the end result is the project deliverables. These are documents which you have created during the course of the project and for which your customer is paying. In most cases your customer does not want to know how you created them or when and in which order you did it, but in the end the documents have to be there. The creation of the deliverables is an iterative process. Although you might see a natural order in which you would create the documents, it is not intended to finalize a document and only then start the next one. If it is reasonable or necessary to change a document because of changes in a subsequent one, then you should create a new revision of the first document. Read on to see what documents typically are part of a project.

### Requirements Analysis Document (RAD)

When starting a project that was initiated by a specific customer request, you will discover that neither you nor your customers know exactly what should be developed by your team or how it should be developed. Your first task will by to read the customer's problem statement (a brief description of the problem). Your team will next need to gather more information about the problem and the desired solution. During this requirements elicitation phase the team should focus on the user's view of the system. The internals (e.g. system design and implementation) and all aspects not directly visible to the user are not part of the requirements. The RAD captures the results of the requirements elicitation phase and is meant to be a 'reference book' for the team during the subsequent development. You can take the RAD to guarantee that design decisions do not run counter to the scenarios defined in the RAD. If some decision violates a goal in the RAD, then a new revision of the RAD should be created.

The Requirements Analysis Document typically contains the

* purpose, scope, objectives and success criteria of the proposed system,
* a description of the customer's current system,
* functional requirements of the proposed system (high-level functionality of the proposed system in natural language),
* nonfunctional and pseudo requirements (e.g. performance, security, hardware platform, etc.), and
* system models (scenarios, use cases, object model and dynamic models describing the proposed system).

For a complete description of how to manage requirements elicitation, see [[1](http://wwwbruegge.informatik.tu-muenchen.de/static/oose/coach/handbook.html#ref1)], chapter 4. For an example template for the RAD, see [[1](http://wwwbruegge.informatik.tu-muenchen.de/static/oose/coach/handbook.html#ref1)] or the [template](http://wwwbruegge.informatik.tu-muenchen.de/static/oose/coach/Documents/Templates/RAD/STARS3_RAD.html) in the accompanying website. There is also an [example RAD](http://wwwbruegge.informatik.tu-muenchen.de/static/oose/coach/Documents/Samples/RAD/RAD_Network.html) from a previous project.

### System Design Document (SDD)

After your team has produced a first revision of the RAD, the focus moves to system design. As with the RAD, the System Design Document (SDD) combines the results of the team's activities during this project phase. The SDD is intended to support the developers during the implementation phase. Ideally, the SDD includes answers to all questions that a developer might have while writing code. Some activities which contribute to the SDD [1] include

* the definition of design goals,
* the decomposition of the system into subsystems,
* the selection of third-party components,
* the hardware-software mapping (which part of the software requires/runs on what hardware),
* a selection of a persistent data management infrastructure,
* an access control policy,
* a design for a global control flow mechanism, and
* finding solutions for handling boundary conditions.

Depending on the number of developers in your project, the project management might have already decomposed the system into subsystems. A different team will then be responsible for each subsystem. A coach will be responsible for each of these teams. If the course size is small (perhaps less than 10 participants) it seems more reasonable to work together on the RAD and SDD and then split the work (i.e. assign each team member to a subsystem) during the implementation phase.

For a detailed description of system design activities see [[1](http://wwwbruegge.informatik.tu-muenchen.de/static/oose/coach/handbook.html#ref1)], chapter 6 and 7. An example template for the SDD can be found in [[1](http://wwwbruegge.informatik.tu-muenchen.de/static/oose/coach/handbook.html#ref1)], chapter 7.5, or in the accompanying website. You can also find an example SDD from a previous project [on the website](http://wwwbruegge.informatik.tu-muenchen.de/static/oose/coach/Documents/Samples/SDD/SDD.html).

### Object Design Document (ODD)

During requirements elicitation the team focused on the user's view of the proposed system. The System Design Document then covered the internal architecture. But both documents did not deal with code, or more precisely, with classes, interfaces, signatures, etc. This last step is made during object design, which includes

* the specification of services (description of class interfaces),
* the selection and integration of third-party components,
* restructuring the object model (e.g. to increase code reuse), and
* object model optimization (e.g. changing algorithms to meet performance requirements).

In cases where there is not much time left to create the ODD, the compiled JavaDoc of the code might be a good starting point (provided the programming language is Java): JavaDoc is a very convenient way of creating an up-to-date specification of the interfaces of each subsystem. Because the ODD in reality often will be created after the implementation, project management and coaches should insist on complete and up-to-date JavaDoc. Without a sufficient ODD, it will be very difficult to understand the hundreds or thousands of lines of code. Chances of successfully reusing the code will drop to zero without this documentation.

The activities during object design are described in detail in [[1](http://wwwbruegge.informatik.tu-muenchen.de/static/oose/coach/handbook.html#ref1)], chapter 8 to 9. Refer to the accompanying website for a [template](http://wwwbruegge.informatik.tu-muenchen.de/static/oose/coach/Documents/Templates/ODD/PAID_ODD.html) and an [example](http://wwwbruegge.informatik.tu-muenchen.de/static/oose/coach/Documents/Samples/ODD/Package-Authentication.ODD.html) of the odd.

### Code of Prototype

The deliverables of the project include a prototype which will be demonstrated to the customer at the end of the project. Often projects last longer than one semester. Therefore a different team might have to continue your work. As mentioned earlier, code reuse depends on the readability and documentation of the code. If the project was successful, even the customer might look at your code. Therefore the coach should keep an eye on the code produced by all team members and make sure they take the time to include meaningful comments.

### Test Manual

Especially in a large project, there probably will be a testing team which provides test stubs and test drivers when implementation starts. Testing activities should be documented as follows:

* the Test Plan documents the scope, approach, resources and schedule of testing activities,
* the Test Case Specification contains the inputs, drivers, stubs and expected outputs of the tests,
* the Test Incident Report records the results of the tests and any differences from the expected output, and
* the Test Summary Report lists all failures which need to be investigated by the developers.

A more detailed description of testing activities and how to document testing can be found in [[1](http://wwwbruegge.informatik.tu-muenchen.de/static/oose/coach/handbook.html#ref1)], chapter 11.

The accompanying website contains a [template](http://wwwbruegge.informatik.tu-muenchen.de/static/oose/coach/Documents/Templates/tm/TM_template.html).

## Typical Project Elements

### Kick-off meeting

The Project Kick-off meeting is the first project meeting and is intended to introduce the students to the project. The project manager should explain the project goals and schedule as well as perhaps presenting some simple use cases. If possible, the project manager should already know some basic facts about the project and have agreed on a list of possible subsystems the final system should have. These ideas should be presented to the students as well.

The students should be given a short introduction on how to sign up for the course in the mechanisms in the selected communication infrastructure, such as e-mail or a registration post to the administration forum (this should include the selection of which team they would prefer to be assigned to, if applicable). However, this should be done after the Kick-off meeting and not during the meeting, as this usually takes a lot of time.

Finally during this initial meeting, the students should introduce themselves and tell the other project members a little about their interests in this project, their current skills and preferences. If there are too many students to do this during the Kick-off Meeting, this can be done instead during the first meeting of each subsystem team.

### Team Meetings

Team Meetings are held regularly (usually once or twice a week) and are the most important meetings for the project. All project decisions are made during Team Meetings by the students (and not by the coaches or project management; a coach or project manager should act as a consultant to the students - see [management roles](http://wwwbruegge.informatik.tu-muenchen.de/static/oose/coach/handbook.html#h2-3)).

A Team Meeting should consist of status reports about the state of the students' current tasks (action items) and of discussion on new issues or issues that need to be decided on. Actions that do not need the team to be present (like writing code or other documents) should not be performed during team meetings. Reviews of these documents, however, should certainly be part of Team Meetings, once the documents close to a final state.

### Management meetings

Management meetings are held once a week. All the coaches and the project management as well meet and discuss what happened in the recent team meetings to keep the project manager and each other up-to-date. Anything that went badly (or not according to plans laid out previously) should be determined. The reasons for the problems as well as ways of resolving the problems should be addressed. Things that went successfully and why should also be discussed. All of these issues should be recorded in the management meeting's minutes.

The coaches should come up with a list of goals to be met during the coming week. This list should include

* items that need to be addressed in each caoch’s coming team meeting,
* any foreseen problems that can be avoided (and what to do to avoid them),
* problems with project progress,
* problems with specific students, and
* any infrastructural issues that need to be dealt with.

### Client visits

Client visits usually are very rare (as the client doesn't have much time) and therefore should be well-planned and well prepared for. A good date for the first client visit is shortly before the RAD (see [deliverables](http://wwwbruegge.informatik.tu-muenchen.de/static/oose/coach/handbook.html#h2-1)) is finished. At this time everyone involved in the project should have enough knowledge to present his or her ideas and to carry on a competent conversation with the client

On the first client visit, you should talk about deliverables and scenarios the team would like to demonstrate during the CAT. You should additionally talk about a date for the next client visit to demonstrate your refined requirements and possibly some parts of the system design.

One of the students should prepare a presentation for a client visit. It should contain at least the state of the project, the goals and scenarios as found so far and your time plan.

Project management will most probably be present at the Client Meeting and will support the students during this meeting. It is very possible that some of the team's ideas about the project will be changed drastically during this meeting.

### Internal reviews

Internal reviews should be held shortly after delivery deadlines for documents and code. These reviews should be planned and announced a reasonable time in advance. The students and coaches should be well-prepared for these meetings.

If code is presented, the internal review should have the character of a CAT. This means that the students should prepare a presentation and have their code in a stable and presentable state.

### Client Acceptance Test

The Client Acceptance Test is the final presentation to the client at the end of the project. It should be well prepared for as the CAT usually determines the success or failure of a project. The students are expected to present slides explaining the system as well as a live demonstration of the developed system. Usually the live demonstration includes the scenarios which client and developers agreed on during one of the first client visits.

## Management Roles

This chapter describes the different management roles that are referred to in the rest of the document and that are needed to successfully complete the project.

### Methodologist

The methodologist is an advice giver: He supports the project with advices and critiques about the processes, project, organization, modeling and the entire course.

### Project manager

The project manager more closely monitors and sets the direction of the project. Both roles, project manager and methodologist, are sometimes referred to as "project management" in this document.

The project manager should attend the weekly management meetings and should always know about the project's current status. He/she is the coach’s first addressee for any questions, problems and/or news. All project members should keep the project manager as well-informed as possible to help him/her make educated decisions.

The project manager has the following repsosibilities:

* Determine objectives, schedule and budgets
* Design software project management plan (SPMP)
* Establish focused and motivated teams
* Determine work procedures, reporting systems and communication infrastructure
* Accomplish project objective within time & budget
* Monitor performance against the plan
* Resolve technical and interpersonal conflicts
* Report project activities to upper management
* Keep the client informed and committed
* Contribute to or describe the team members performance approval.

### Coach

The Coach is the Developer's consultant. He/she gives advice and ideas to the Developer (the student), but should try to avoid decisions that could instead be made by the team.

The Coach should keep a close eye on the working of his/her team. If something seems to be going wrong, he/she should alert the team to the problem but then let the team decide on a solution. In case of team member conflicts, he should help to resolve them if they cannot be resolved otherwise.

Any project experience that the coach possesses should be imparted whenever possible to the students. Toward the beginning of the project, the coach will probably have to lead his/her Developers more than in the later phases of the project (see Team Meeting). The coach is responsible for the Developers on his/her team, and should try to keep the team members inspired and willing to fulfill their part.

Above all, the coach should keep in mind that the students are in the course to learn. This is the first and foremost goal of the project. If a coach has to choose between a more successful project and a more educational project, the choice should always be the educational project. It may at times be frustrating for a coach to watch teams make decisions that are obviously (to the coach) inferior. However this is an essential part of the course; all effort should be made to let the students make decisions on their own, about every aspect of the course.

Some tasks of the coach may be:

* To delegate tasks if no volunteers exist. This should be less necessary as the course progresses; the team leader should handle delegation of work.
* To keep in mind the project goals and push discussions and agendas towards fulfilling these goals. This should involve understanding the RAD and assuring that all requirements specified in this document are being met.
* To make sure the team is aware of significant dates.
* To support and advise the Minute Taker and Primary Facilitator. The Primary Facilitator will likely need assistance in the beginning of the course to identify what work the team needs to focus on. As the project progresses, it will be clearer to the team members what they need to do.
* To review presentations and documents. This should be for two main purposes. First, the coach can provide feedback while the documents are being generated. Second, is the coach's responsibility to give grades and provide comments about presentations or submissions.
* To try to involve all team members. Participation in the project is a requirement for every student. If a student does not seem to be actively participating, maybe he or she just needs a little help to find a way to contribute.
* To schedule rooms and equipment, or to help with equipment issues. The coach should provide assistance in dealing with the details of reserving facilities or requesting support for hardware/software.

### ‘Developer

The Developer is a student; usually a junior or senior in the Computer Science department or the Electrical and Computer Engineering department. In general, this student does not have any software development project experience. However, the student should be attending the software engineering lectures and should be excited about the project and be willing to learn more about software engineering.

### Technical Writer

The Technical Writer is a Developer who has agreed to be responsible for a special document. This does not mean that the Technical Writer has to write this document on his/her own, but means that the Technical Writer has to keep track of the document's state, assign specific tasks to other students and assemble the parts he/she received into a single structured document.

The Technical writer should always know who still has action items related to the document he/she is assigned and should know the state of this action item. If the Technical Writer assigned a task to the wrong person by mistake or if an assigned person is not capable of fulfilling the task, the Technical Writer has to re-assign that task to another person.

### Client

The Client is the "customer" of the project. Generally the client is a company prepared to spend money on the result of the project (if it is successful). The success of the project is measured by how well it fulfills the client's requirements. This does not mean that the client knows exactly what those requirements are. In fact, the client rarely knows exactly what the project requirements are or can express them clearly, hence the "requirements elicitation" phase of the project.

# Meetings

Now that you have been presented with project management basics, this section deals with one of the most significant elements of project management: the meetings. Large groups can't work together without meetings. Without meetings, nobody knows what other team members are doing. The team will be unable to solve complex problems without this collaboration. This section first describes the different types of meetings, then talks about the different roles that team members must assume in meetings. Finally you are provided with outlines of some sample meetings and other information that will help to make your meetings more effective.

## Types of meetings

A normal meeting will probably not be one of these meeting types but will consist of different phases with each phase being one of the types mentioned in this chapter. Usually, you share information at the beginning of the meeting (see "Information only"), then try to come up with solutions for new problems (see "Brainstorming") and then try to decide about open issues (see "Decisive").

### Information only

An Informational meeting does not resolve any open issues. It informs the team of facts or updates the group's knowledge. Examples of topics in informational meetings are tutorials and status updates. The information is usually delivered by a single team member or by an outside source (e.g. a visiting member of a different team).

### Brainstorming

A Brainstorming meeting does not resolve any open issues either. If the team needs to come up with new ideas on how to deal with a specific problem (e.g. for finding subsystems, objects, problem solutions, etc.), the team can hold a brainstorming session.

During Brainstorming, possible solutions are only written down and should not be criticized, weighted or talked about. This should be done after the brainstorming session.

### Decisive

In a Decisive meeting, the team has to resolve open issues, argue with pro and con arguments about solutions for these issues (decisions and resolved issues dominate the meeting). Each issue should be addressed one after another and not all at the same time.

## Meeting Roles

In this section we will describe the different roles that can or should be filled during meetings. Please make sure that in a meeting, all members agree upon who is assigned which role(s) before the meeting is started.

### Primary Facilitator

The Facilitator is acting as a neutral servant of the other group members. The primary responsibilities of the meeting facilitator are to

* assure that all roles are clearly defined and agreed upon,
* maintain a common focus on content (what should be discussed?) and on process (how should it be discussed?),
* guide the group (focus the energies of the group) without directing it,
* use group decision making to earn each member's support for final decision,
* maintain an open and balanced conversational flow,
* make sure that all participants have an opportunity to participate and that none dominate the meeting, and
* protect individuals from personal attack and knock down walls which have been built between people.

When the group is working well together, the Facilitator should not need to do much and can let group members speak spontaneously. If things become heated or bogged down, the Facilitator should step in and more directly control the , signaling who should speak next, cutting off aggressive behavior, and keeping the group to its agreed-upon task.

A Primary Facilitator should expect and receive facilitation assistance (ideas on what could be done next) from everyone in the group (so-called secondary facilitation). However, the Facilitator always should have the ability to pick the idea he/she likes best or to come up with his/her own idea on how to continue facilitation.

Ideally, the process role (the facilitator) should be separate from the decision-making role. In other words, bosses shouldn't run their own meetings. In software engineering meetings, everyone not being the facilitator is in the decision-making role. If the facilitator wants to contribute to the discussion, he/she should signal that he/she is temporarily stepping out of the role of facilitator. When done, he/she should announce the return to the role of primary facilitator and proceed as facilitator. Secondary facilitation (all other meeting members) should always make sure that the primary facilitator doesn't switch roles unconsciously.

### Recorder

This role is optional and is not needed at every meeting (a Recorder is most commonly used during brainstorming meetings). It may not be necessary to have a Recorder for the whole meeting; only for specific agenda points during a meeting.

The Recorder keeps track of what is said in the group and writes it down on flip chart pages or a whiteboard visible to everyone in the group. This is called the short term group memory. Please note that the recorder never runs the meeting. This is the primary facilitator's task. The recorder does not interpret, edit or in any way alter what is said when recording it.

The Recorder may lag behind the conversation. However, if the Recorder gets too far behind, he/she should ask the group to stop for a moment, so he/she can catch up.

### Minute Taker

The Minute Taker takes notes concerning decisions reached and action item assignments (who has agreed to do what by what date). This information should be made public to the other members in a tool suite supporting the communication and documentation needs of the project (as a response to the Agenda) within 24 hours of the meeting. These minutes serve as a long-term group memory. They are extremely important not just to remind group members what they talked about during a meeting, but to let members of other groups or other projects track what is being discussed and decided.

The selection of the tool suite is specific to the environment of the project, university or company. In the past we have set up communication infrastructures based on proprietary solutions such as Lotus Notes or with an open source solution such as pbhBB. For the CASE tool we use products supporting the whole software development lifecycle such Rationale Rose as well as drawing tools supporting only the drawings of UML diagrams such as OmniGraffle. Currently we are using a tool, which unifies the communication, modeling and project management issues, the open source tool Unicase [5].

### Time Keeper

The Time Keeper monitors how long the group is taking to accomplish its tasks and provides regular updates to keep members aware of where they are with regard to time spent. The Time Keeper should always speak up if he/she thinks that the group won't finish at the scheduled time.

If the allotted time has been used and the desired outcome is not close to being achieved, the group needs to decide whether to continue processing the current topic to its conclusion or stop and move on to the next item.

## Sample Meetings

Because of the importance of the first few meetings, we will describe these meetings so that you know what to expect from them and your role in each.

### Kick-off Meeting

Usually this is the first meeting that will be held during the course of the project. The kick-off meeting will be conducted by the project manager(s). The expected audience is all project members including all coaches. This meeting should give the students a first idea of what the project is intended to accomplish. The customer's problem statement should be presented. This is a description in natural language of what the customer expects the prototype to do at the client acceptance test. This might contain scenarios to help explain these ideas. The kick-off meeting also should answer organizational questions like where and how to register for the project, how to get a key for the lab, an account for the computers, etc.

### First Management Meeting

The first management meeting should take place before the first team meeting and as you might have guessed, the attendees are all coaches and the project manager(s). The purpose of the meeting is to prepare the coaches for their tasks. The coaches should come away from the project with an understanding of the project management basics (see chapter 2) and how meetings will work (see chapter 3). Normally, coaches do not act as primary facilitators during team meetings. But the first team meeting is an exception to that rule. Coaches will also have to teach the team members in their group how to become a good primary facilitator. So take the first management meeting as a chance to learn the role of a primary facilitator and feel free to ask questions about that. It also might be a good idea to discuss some sample agendas during the first management meeting. During the first management meeting and also during the regular management meetings, you should discuss what tasks the students should accomplish during the next week. As an coach you are responsible for keeping your team informed of the steps laid out during the management meeting.

### First Team Meeting

Only the team members and a coach will attend the first team meeting. The first team meeting will be held after the kick-off and the first management meeting and, of course, before the first regular team meeting. Because this is an introductory meeting, you will be the primary facilitator. Therefore you will provide the agenda which not surprisingly will primarily contain organizational tasks. Some topics you might include in your agenda:

* Meet the group members:

Let the team members introduce themselves. Introduce yourself as well. Have students discuss what they expect from the course.

* Assign a minute taker:

Because organizational issues will be discussed during this meeting, it is important that someone takes notes and creates a summary after the meeting. Assign this role to a team member and briefly explain what he or she is expected to do.

* Make sure the infrastructure works

Has everybody registered for the course? Does everybody know how to get an account for the computers, tools and forums? How about keys to the lab? Also introduce the different rooms and how to get there.

* Deliverables:

Discuss with the team members what they are expected to deliver when the project is over (see 2.1).

* How meetings work

Explain the meeting roles and what people are expected to do when they hold a role (see 3.2).

* Meeting dates and role assignment:

Work out a schedule for the whole project which contains meeting dates and the names of the team members who will hold a role. Each team member should hold the role of primary facilitator at some point during the project. If no one volunteers for the roles, be prepared to assign roles yourself. Remind the minute taker to post the minutes of the meeting, including date, time and location of the next meeting and the names of the students who will hold roles in that meeting.

* Members' understandings the problem statement:

You might include a short discussion about the objectives of the project as they are seen by the team members after the kick-off meeting. To avoid confusion and frustration about the usually vague problem statement, point out that finding a clear definition of the problem will be the task of the next weeks ("requirements elicitation phase").

* The next steps:

You might already be able to present a rough schedule for the next couple of meetings, e.g. next meeting working on scenarios, the meeting after that transforming scenarios to use cases and then preparing the first draft of the RAD.

### Meeting with Student Primary Facilitator

When a student becomes primary facilitator for the first time, the coach should meet with him/her at the latest two days before the next team meeting. The purpose of this face-to-face meeting is to individually teach the student how to become a good facilitator. The coach should give some pointers on how to make meetings work (see 3.4) and how to deal with problem people (see 3.5). It is also a good time to jointly develop ideas for the next agenda (ideally the meeting takes place after the management meeting so that the coach knows which steps in the project will come next). The student still should have time to post the complete agenda 24 hours before the team meeting.

After every team member has been primary facilitator the face-to-face meetings should not be necessary any more. Then the coach should suggest topics for the agenda via forum.

### First Regular Team Meeting

The first regular team meeting takes place after the first management meeting with the first student primary facilitator. All group members and the coach are expected to be present. Ideally, the meeting should be completely conducted by the students. The primary facilitator works through the agenda, the recorder writes down the group memory on the whiteboard, the minute taker takes notes for the group's long term memory and the time keeper helps to finish the topics in time. The coach should support the primary facilitator, the recorder and minute taker to assure that no idea or solution is lost. If nobody volunteers for an action item, the coach has to assign the task. Be sure the minute taker writes down both the name and the date when the task should be finished. If a discussion becomes irrelevant or to lengthy, the coach should intervene and bring the group back to the point. To discover problem people, the minute taker should write down missing students or even those who came late.

## How to make Meetings Work

Your meetings will not always run smoothly. In this chapter, you'll find some hints about how to improve your meetings, starting with how to judge a meeting (what is a good meeting?) and ending with some common problems with facilitation and how to solve them.

### Judging a meeting

There are two ways of judging a meeting and both of them are important and should be used equally. The first is judging the results: what happened during the meeting? Did the team get their tasks/work done? Do the results meet your requirements? Are the ideas innovative?

The second is judging how well the group worked together. Was everyone helping the team to get the results? Did everyone participate? Did the group work together well? Any hard feelings? Was the meeting enjoyable?

### Group memory

Each person has a short term memory and a long term memory. The group meeting should have these kinds of memory too. The long term group memory takes the form of the minutes. The short term group memory is the Recorder's notes. These notes are written down during the meeting and visible to the whole group (e.g. like on large sheets of papers or on a whiteboard). This makes it possible for every group member to check whether the group memory correctly reflects his/her ideas. As the short term group memory is the basis for the minutes, this dramatically improves the quality of the minutes as well.

### Things that can go wrong and how to solve them

**The group needs the Facilitator's help during a conversation**

Here is a small extract from a meeting with everything going wrong that can go wrong:

|  |  |
| --- | --- |
| John: | We're supposed to provide a list with what is going wrong in this lab. |
| Oliver: | The NetBoot environment is slow and unstable. |
| Chris: | Why do you always have to complain about our Macintosh computers? We've been having this problem for over a year now and it can't be done any better, so shut up. |
| Susan: | If we would buy a new switch, we perhaps could reduce that problem. |
| Chris: | I don't think that helps. |

  Here's how the meeting could go if the facilitator detects all problems and knows how to solve them:

|  |  |
| --- | --- |
| John: | We're supposed to provide a list about what is going wrong in this lab. |
|  | (John only provides a content-focus, but no process-focus on how to get there) |
| Facilitator: | Let's try the brainstorming method here. Everyone tries to bring in his or her ideas and we try not to judge them until we have found enough statements. |
| Oliver: | The NetBoot environment is slow and unstable. |
|  | (On a difficult problem, the first person who speaks up should be rewarded. This encourages other members to bring in their ideas and discourages interrupters) |
| Facilitator: | Very good, Oliver. Anybody else? |
| Chris: | Why do you always have to complain about our Macintosh computers? We've been having this problem for over a year now and it can't be done any better, so shut up. |
|  | (Chris is not supportive and talks against Oliver taking his statement as an personal offense) |
| Facilitator: | Chris, we agreed on not to judge each other's statements. What do you think is going wrong? |
| Chris: | There's a link broken, so the Compiler sometimes crashes. |
| Facilitator: | Thank you, Chris. Any other ideas? |
| Elaine: | Yesterday, I lost a java source file! |
| Susan: | As if anyone cares. |
|  | (Elaine brings in personal problems being (mostly) off-topic; Susan criticizes Elaine in a very rude way. The Facilitator has to show sympathy for personal problems but has to try to keep the focus the content) |
| Facilitator: | Susan, let's try not to attack each other. Elaine, that must have been extremely annoying for you. Perhaps you could rephrase your statement and tell us in a small sentence what the problem was. |
| Elaine: | The compiler sometimes deletes the input files. |
| Chris: | That's the broken link problem. |
|  | (Chris tries to solve problems while in brainstorming phase) |
| Facilitator: | Sorry Chris, we will try to evaluate and solve the problems after our brainstorming phase. |

**Facilitator needs to avert the "jumping in" problem**

Jumping into a conversation can be difficult and stressful, because you have to remember your statement until you find a chance to jump in. In most cases, this will be distracting enough that you won't be listening to your colleagues. This is why a system is needed to get everyone talking without this effort (like a policeman regulates traffic on a crossing, the Facilitator should regulate the discussion).

The group sometimes won't be very supportive when someone "jumps into conversion". In this case, the facilitator should try to thank that person for his/her comment and try to encourage him/her to come up with further statements, even if the first one was inappropriate.

Another hint for all group members: if you come up with something important during a meeting, but cannot bring in your idea at the current point of conversation, write down your idea so you are not distracted with remembering it.

**Same problem, different solutions**

Another problem is the following where John's introduction leads the group in the wrong direction:

|  |  |
| --- | --- |
| John: | It's clear to me our problem is that we have too many subsystems bloating our system. How about removing the Logging and the Security subsystem? |
| Robert: | Those subsystems are important. We won't have enough features if we start removing subsystems. We should rather profile our current system. |
| Sylvia: | What about removing just one subsystem? |
| Robert: | I know a very good profiler for Java. |
| Martin: | What would happen if we'd combine two or more small subsystems and build a larger subsystem? |
| Angela: | I heard there will be double capacity memory cards in a few months... |

  The problem was not presented as a question about how to solve it (in this case: "Our system is too large for our wearable computer. How can we reduce the memory usage?"). Instead a preconceived solution was presented ("we need to remove subsystems"). This can cause everyone else to come up with their own ideas without listening to each other and the meeting will lead nowhere.

## How to deal with problem people

Regardless how well you prepare a meeting there may always be some team members who disturb the meeting through their behavior. It is primarily the task of the facilitator to make the meeting efficient, i.e. to avoid disturbances, but the coach or project manager should support the facilitator. Some of the most typical types of 'problem people' are shown here together with some thoughts about how to stop the unwanted behavior (also see [[2](http://wwwbruegge.informatik.tu-muenchen.de/static/oose/coach/handbook.html#ref2)], pp. 107-117).

### The Latecomer

**Problem:** The latecomer interrupts the meeting by rushing in the door. Even more disturbing, he/she wants to be caught up forcing the facilitator or some other team member to repeat the whole meeting since the beginning. Because all other people know what has been discussed they get the feeling that this repetition is a waste of time.

**Possible solution:** Always start the meeting on time. By waiting for late team members the other people in your group will start adjusting to the new meeting time and also arrive late. Ask the latecomer after the meeting for the reason of the delay. You can also ask him/her what would make the meeting important enough to want to be on time. Don't waste other people's time by briefing the latecomer during the meeting. Let him/her catch up by reading the group memory (whiteboard).

### The Early Leaver

**Problem:** The early leaver distracts other people from focusing on the meeting and "drains the energy" of the meeting.

**Possible solution:** Find out later why he/she left early. Maybe meetings are too long or do not finish on time. At the beginning of the meeting ask all people if they can stay till the end. It is much less probable that someone leaves early when everybody agreed to stay till the end of the meeting. If someone has to leave early, decide if the meeting should continue anyway. Do not continue if several people have to leave.

### The Broken Record

**Problem:** Some team member keeps bringing up the some point over and over again.

**Possible solution:** Use the group memory (whiteboard) to acknowledge that the point is important to the individual. Demonstrate that it has been heard and recorded several times.

### The Doubting Thomas

**Problem:** He or she is always negative and criticizes everything ("That will never work.", "That will never happen.", "I don't like that.")

**Possible solution:** Try to get the group to agree to a process of not evaluating ideas during a brainstorming phase. Use this agreement to correct the Doubting Thomas when he does nevertheless. Try to establish the idea of constructive criticism: If someone does not like something, he or she has to give a better alternative.

### The Head shaker

**Problem:** Disturbs the meeting in a nonverbal way by shaking the head, rolling eyes, slamming books shut, pushing chairs back, etc.

**Possible solution:** Try to ignore the head shaker and focus on the person who speaks. If it is too annoying, speak to the head shaker as if he/she had made a verbal comment. Make the head shaker aware of his/her body language, e.g. by asking why he/she had rolled his/her eyes.

### The Dropout

**Problem:** The dropout sits at the back of the room and does not say anything or even reads a book.

**Possible solution:** Try to bring the dropout in again by asking him/her a question but without embarrassing him/her: "Frank, what's your opinion? Maybe you want to think about that for a second. Susan, how about you?" Maybe there is a reason for the person to drop out: the topic could be irrelevant or there is no reason why the person is at the meeting at all.

### The Whisperer

**Problem:** Someone constantly whispers to a neighbor.

**Possible solution:** As facilitator/coach/project manager if you are standing, you might just walk up closer and closer to the whisperer. Often just this physical presence will work. If not, remind the group in general to keep a single focus. If it is still going on, ask the whisperers to share their point of view with the group. If they still continue, you might ask them to go outside the meeting room for their talking. Next meeting you could find a way to get chronic whisperers to sit apart from each other.

### The Loudmouth

**Problem:** The loudmouth talks to much and too loud and dominates the meeting.

**Possible solution:** As with the whisperers, if you are standing try to move closer and closer to the person until you are standing right in front of the loudmouth. Often this will make the person aware of his/her behavior and he/she will stop talking. Then immediately shift your focus and call someone else. If the loudmouth is someone who has to blurt out ideas as soon as they come, give him/her a pad of paper and ask him/her to create a personal memory (i.e. writing down ideas). You also could ask the person to serve as a recorder in order to keep him/her busy.

### The Attacker

**Problem:** Someone launches personal attacks against other team members or the facilitator or coach or project manager.

**Possible solution:** Remind the group that everyone is at the meeting to work on a task, not to watch some people working out their personal problems. Use the whiteboard to refocus on ideas rather than individuals. You can also try the technique of deferred evaluation (also see "The Doubting Thomas", 4.5.4). If it is you who is being criticized, try to resist to defend yourself. Instead, thank the attacker for his/her criticism, and then turn the issue back to the attacker and ask him/her how you should have done it better. Also try to involve the group and ask the other team members for their opinion.

### The Interpreter

**Problem:** A team member always speaks for other people: "What Michael is trying to say..."

**Possible solution:** Stop the interpreter immediately if he/she is interrupting another team member: "Please let Michael speak for himself. Go on Michael, finish what you were saying." If Michael has already finished, check the interpreter's understanding. "Did George understand what you were saying?" With this technique, it will become less likely that someone tries to act as a mouthpiece of someone else.

### The Gossiper

**Problem:** The gossiper introduces hearsay, gossip or vague information into the meeting ("Somebody mentioned that..."). Discussion whether this information is accurate can waste a fair amount of meeting time.

**Possible solution:** If it is important information, try to verify it immediately. Either ask the other team members if they can confirm the information or create an action item for someone who has to investigate the issue until the next meeting.

### The Know-It-All

**Problem:** Someone uses his/her experience or practical training to argue a point: "I've been working on this for a whole semester and I know that this cannot be a solution for our problem."

**Possible solution:** Emphasize why this issue is being considered by the group: to come up with new insights and ideas. Ask the know-it-all to hold back for a while and to let the group gather some ideas.

### The Backseat Driver

**Problem:** A team member keeps telling you what you should be doing: "I would move on the next issue if I were you."; "Tell him to shut up."

**Possible solution:** If someone suggests a different process you should ask the other team members if they agree and then follow the suggestion. If the backseat driver permanently criticizes the facilitator, ask him/her to if he/she wants to continue as a facilitator if the group agrees. If the backseat driver does a worse job as a facilitator, the group should ask the original facilitator to step in again.

### The Busybody

**Problem:** Someone repeatedly rushes in and out the meeting because of phone calls, important e-mails he/she is waiting for or because of appointments with someone else who might or might not be important to the project.

**Possible solution:** Try to reach a commitment by all team members to switch off all cell phones before the meeting and not to do the homework (e.g. searching the internet, sending and reading e-mails) during a meeting. As a coach or project manager be a "shining example".

### The Interrupter

**Problem:** Some team member does not have the patience to let other people keep talking until they are finished.

**Possible solution:** Like the loudmouth (see 4.5.8) the interrupter might be afraid that he/she could lose an idea if it is not blurted out immediately. The solution is quite similar to the loudmouth solution: Remind the person to be patient and to let other people finish their thoughts. Ask him/her to bring a pad of paper and to create a personal memory or suggest that he/she becomes recorder.

### The Teacher's Pet

**Problem:** A team member spends more energy looking for approval from you than focusing on the content of the meeting.

**Possible solution:** If a teacher's pet keeps talking to you rather than other group members, try to break eye contact and involve other team members. "I don't know. How do you think the meeting's going, Shalin?" The idea is to get people talk to each other, not to you.

# References

[1] Bernd Bruegge, Allen H. Dutoit: "Object-Oriented Software Engineering: Using UML, Patterns, and Java", 3rd Edition, Prentice Hall, Upper Saddle River, NJ, 2009

[2] Michael Doyle, David Straus: "How to make meetings work", The Berkley Publishing Group, New York, NY, 1982

[3] Thomas A. Kayser: "Mining group gold", Serif Publishing, El Segundo, CA, 1990

[4] Watts S. Humphrey, et al: "Introduction to the Team Software Process", Addison-Wesley Professional, 1999

[5] Unicase