

# Practices for Involving Stakeholders

PLEASE SIT IN TEAMS

**Ann Majchrzak**

**September 21, 2001**

**Marshall School of Business**

**University of Southern California**

Marshall



McCOMBS SCHOOL OF BUSINESS  
The University of Texas at Austin



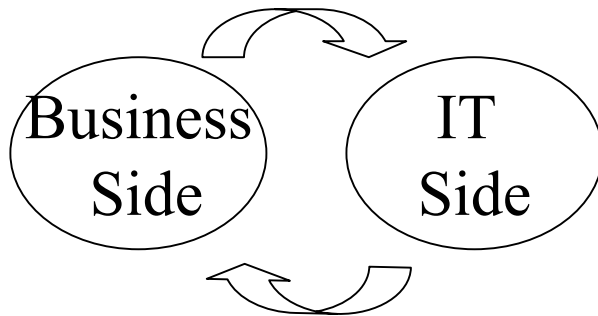
# Purpose



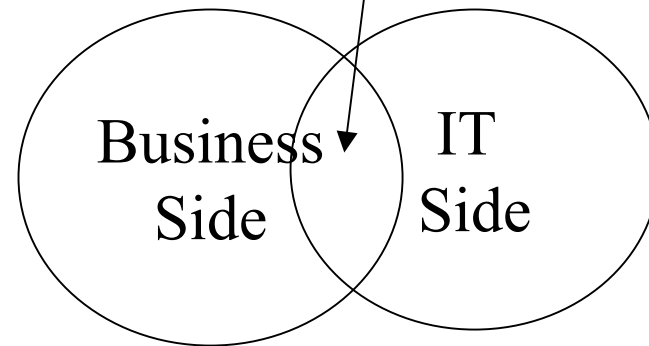
**To inform you of practices for involving stakeholders (clients & developers) in a “collaborative learning process” to achieve innovative business-IT solutions**

# What is “Collaborative Learning”?

This is Knowledge Transfer or Individual Learning:



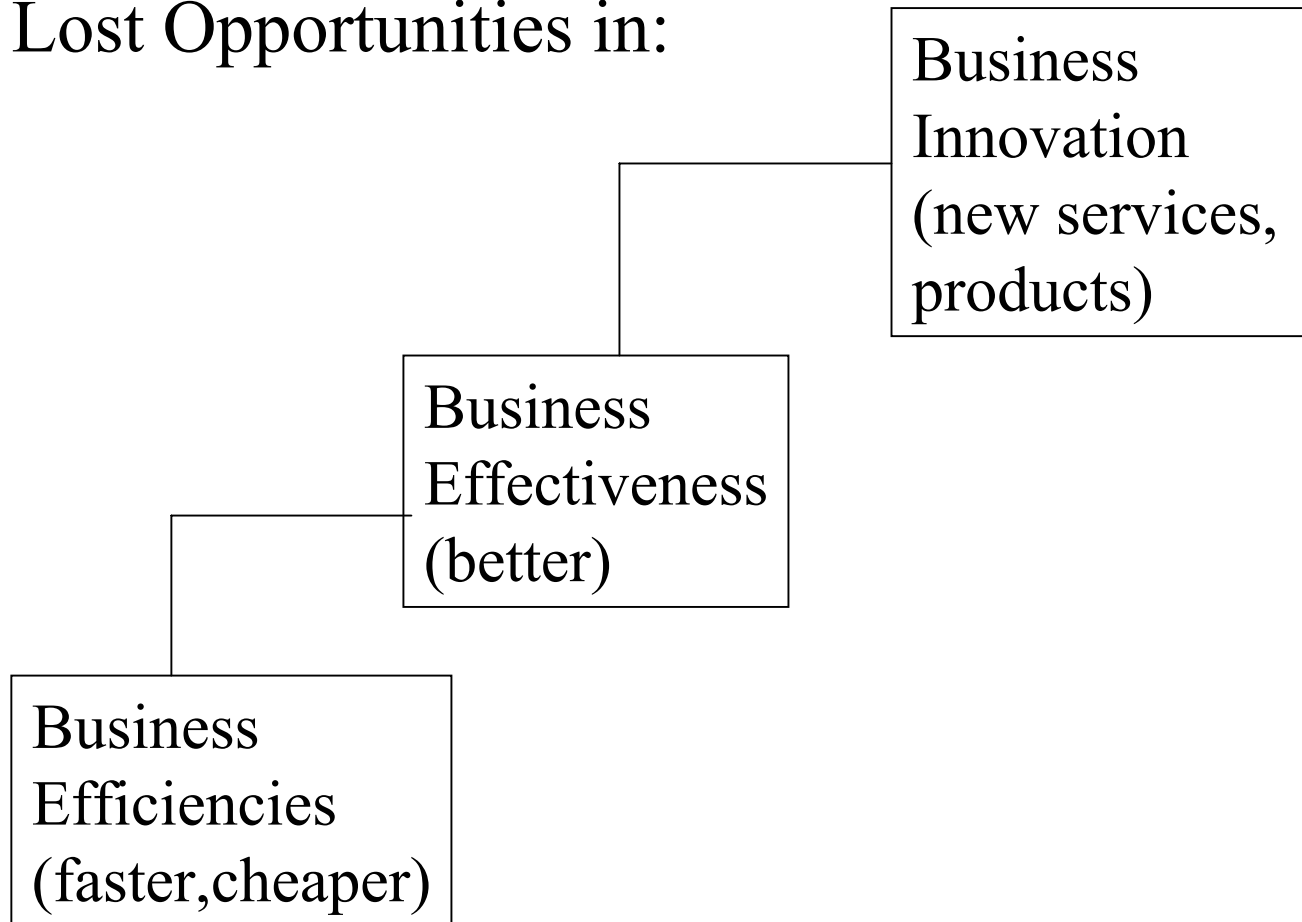
This is Collaborative Learning



Together, learning new ways of structuring IT and business processes

# Why Worry about Collaborative Learning?

Lost Opportunities in:

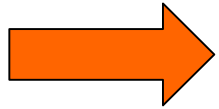


# Recent Findings from Local SW development firms



	High Learning Stakeholders	Low Learning Stakeholders
Average use of Collaborative Learning Techniques	4.5	1.2

# Practices for Encouraging Collaborative Learning



- I. Creating Shared Responsibility
- II. Elaboration
- III. Managing Conflict to achieve learning

# **I. Shared Responsibility**



**What is it? It is the psychological attitude that “we’re all on the same team”; “we’re in this together”**

# Practices for Encouraging Shared Responsibility:

## When Managing Stakeholder Relationships

Help to make ALL stakeholders part of development team:

- ☒ Put on email distribution list
- ☒ Include in teleconferences
- ☒ Frequent interactions, even if quick
- ☒ Identify tasks that developer and client can work on together
- ☒ Use “we” not “I” during discussions
- ☒ Identify team-based rewards (such as lunches)
- ☒ Define project success metric as system use, not just system development

# Practices for Encouraging Shared Responsibility

when Project Starts

- 1) Agree that learning should be part of development effort
- 2) Identify learning objectives for each stakeholder:
  - Client organization's work process
  - IS development process
  - Technology developments
  - Use of IS in business
- 3) Give priority to overlapping learning objectives
- 4) Identify & Assign development tasks related to each learning objective

# **Practices for Encouraging Shared Responsibility**

## When Managing Meetings

- 1) Appoint a Learning Facilitator
- 2) Start each meeting with learning objectives
- 3) Identify & assign tasks based on learning objectives
- 4) End each meeting with assessment of learning

# Exercise



- ⌘ **Discuss each person's learning objectives**
- ⌘ **Prioritize overlapping objectives**
- ⌘ **Identify & assign development tasks related to objectives**
- ⌘ **Create a form documenting decisions**
- ⌘ **Appoint learning facilitator for meetings**

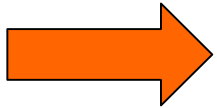
# Practices for Encouraging Collaborative Learning



I. Creating Shared Responsibility

II. Elaboration

III. Managing Conflict to achieve learning



# Purpose of Elaboration

Create a common body of knowledge shared by stakeholders about:

- ☒ A vision of the IT-enabled to-be work process
- ☒ Business & technical rationale for vision compared to alternatives
- ☒ Execution Plan
- ☒ Goals, preferences, and fallback options for each stakeholder ("What does each what to accomplish? What happens if it doesn't work?")
- ☒ The most efficient ways for each stakeholder to learn

# Elaboration Practices



- 1) Focus on Actual Work Processes, not hypothetical ideal (to identify new areas of improvement, ex: rework cycles) by:
  - ⌘ Observing clients doing work
  - ⌘ Identifying decisions not just actions
  - ⌘ Observe users role playing (or actually) using prototypes
  - ⌘ Keeping focused on business objectives of work process (not of system)

# Elaboration Practices



2) Learning is not telling; self-explainers learn more than listeners

Thus, allow time for everyone to “self-explain”

# Elaboration Practices



- 3) Make abstract discussions concrete
  - ⌘ Use examples related to client's work
  - ⌘ Diagram ideas
  - ⌘ Build on the client's methods for describing work, not own techniques (like "ERD")
  - ⌘ Translate terms
  - ⌘ Show other IS's as examples

# Elaboration Practices



## 4) Customize Learning Techniques

- ⌘ Try different techniques
- ⌘ Ask frequently: are you learning what you wanted to learn?

# Elaboration Practices

## 5) Keep Creative Ideas Flowing:

- ⌘ Quickly create & discard lo-tech prototypes to explore concepts (not as status assessments)
- ⌘ Role play use of prototype in alternative to-be work processes
- ⌘ Stimulate creative discussions with:
  - ☑ "What would happen if...?"
  - ☑ "Had you thought about ...?"
  - ☑ "If we did X, what would happen?"
  - ☑ "What are strengths & weaknesses of...?"

# Elaboration Practices



## 6) Active Listening

- ⌘ Ask about unstated reactions to idea
- ⌘ Switch roles
- ⌘ Avoid talking too much
- ⌘ Restate what you heard
- ⌘ Build on the client's examples & ideas

# Exercise:



**Conduct a 10-minute design meeting and see how many Elaboration techniques you used (e.g):**

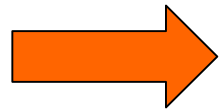
- \*\*\* Observe users roleplay with prototypes**
- \*\*\* Time to self-explain**
- \*\*\* Concrete examples**
- \*\*\* Translate terms**
- \*\*\* “Are you learning what you wanted?”**
- \*\*\* Role play alternatives**
- \*\*\* “What would happen if...?”**
- \*\*\* Ask about unstated reactions**
- \*\*\* Build on others’ examples**

# Practices for Encouraging Collaborative Learning



I. Creating Shared Responsibility

II. Elaboration



III. Managing Conflict to achieve learning

# How can conflict help learning?

	Client	Developer
Goals (time, resources, outcomes)		
FallBack Options		

**Which Alternatives meet these?**

# Problems with Managing Conflict



- ⌘ Most people aren't fully conscious of their preferences or their fall back options prior to the moment of conflict.
- ⌘ During conflict, the last thing anyone wants to disclose is preferences or fall back option

# Practices for Managing Conflict to achieve learning



- 1) Develop Goal Hierarchies (which goals are more important; which are shared?)
- 1) Start identifying alternatives and compare them to Fallback Options and shared goals (Don't focus on one alternative yet)

# Exercise: Generate:

Own

Others

1) Goals for:

- time
- resources
- SW outcomes
- Other outcomes  
(learning?)

2) Goal Hierarchy & sharing

3) Fallback options

4) Pick a solution, evaluate against Fallback

# Ex Difference in Practices: Individual vs Collaborative

- ⌘ Use prototypes for single solution
- ⌘ Enforce single representation of knowledge (“ERD”)
- ⌘ Explain own knowledge
- ⌘ Talk
- ⌘ Stay in role

- ⌘ Use prototypes to explore different concepts
- ⌘ Represent knowledge in different ways
- ⌘ Have others explain your knowledge
- ⌘ Draw, listen, ask questions
- ⌘ Reverse roles

# Checklist during meetings



Did you?

- ☒ Use prototypes to explore concepts?
- ☒ Let clients develop prototypes
- ☒ Create “test-drivable” prototypes?
- ☒ Make sure client asked as many questions as you did?
- ☒ Stimulate creativity through questioning?
- ☒ Restate dialogue to improve understanding?
- ☒ Use examples from more than one work context?
- ☒ Avoid using IT-language?

# Checklist during meetings (Cont)



Did you:

- Use visual examples to explain concepts?
- Reversed roles?
- Tried more than one way to represent how work is done?
- Elaborated on client's idea?
- Grounded ideas in client's physical world with a role play by sharing stories of how work is done?
- Asked about client's unstated reactions to an idea?
- Did you show any IS's that client might want to emulate?

# Summary



- 📄 Every client-developer encounter is an opportunity for learning
- 📄 Every client and developer learns differently
- 📄 Controlling the learning process is better than leaving it uncontrolled
- 📄 Control it by:
  - 📄 Building and maintaining a sense of shared responsibility for outcomes
  - 📄 Use elaboration techniques
  - 📄 Manage conflict for learning



**Thank you!**

Students will be receiving additional training in these techniques late in the semester outside of class

# Role Play



Client: You made the help key F1 and I want it to be F12. How soon can you change that?

Dvlpr: It's no problem, but WHY do you want it to be F12?

Client: Help was F12 on our old system, and I think it would be easier for everyone to learn this system if it was the same. I want them to get used to getting help from the system instead of calling you.

Dvlpr: Well, F1 is the standard for help on almost all systems, and we'd like them to get used to using F1, since that is where help will be on all their other systems. I guess the principle we're both after is how to minimize transition costs? Is that right?

Client: Yes, that's what I was thinking about

Dvlpr: And also we're in total agreement that we don't want them relying too much on the support staff!

Client: Absolutely!

Dvlpr: Well, let's see. How about if we make both F1 and F12 help keys? That way all the people used to the old system could go right to the F12, and any new people you hire, who are used to F1, would find Help there. Would that work for you? It would be easy for us to change. Or do you have another suggestion about how to minimize transition costs?

Client: I guess another way would be to display a message with F12 that said "HELP IS AT F1." That way they would all gradually learn to use F1, wouldn't they? After a while we could use F12 for something else, then, couldn't we? But which solution will work the best?

Dvlpr: I think that using both F12 and F1 would minimize transition time and long term support costs.

Client: Great, let's go with that. I know you've got better things to do with your time.