SOFTWARE ENGINEERING

Week 1

Introduction – Software Projects

Asst. Prof. Dr. A. Cüneyd TANTUĞ   Asst. Prof. Dr. Tolga OVATMAN
Istanbul Technical University
Computer Engineering Department

Course - Introduction

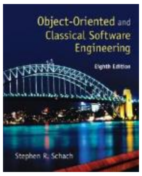
- BLG411E – Software Engineering
  - İTÜ Credits : 3-0-0
  - ECTS Credits : 7,5
  - Course Web Site : www.ninova.itu.edu.tr
    - Lecture notes, announcements, report templates and examples, tools, etc.

2015-2016 Fall Semester		
	CRN	CRN
Instructor	Asst.Prof.Dr.A.Cüneyd TANTUĞ www.tantug.com	Asst.Prof.Dr.Tolga OVATMAN www.ovatman.com
Office	3309	5309
Office Hours	Tuesday, 09:30-12:30	Tuesday, 09:30-12:30
Teaching Assistant	R.A. Bilge Süheyla AKKOCA R.A. Müge Erel R.A. Beyza Eken	

Course - Textbooks

Software Engineering:
A Practitioner's Approach (SEPA)
Roger S. Pressman, 6th ed.
McGraw-Hill, 2005

Object-Oriented and Classical Software Engineering
Stephen R. Schach, 8th ed.
McGraw-Hill, 2010



Course - Supplementaries

Other Books	• Software Engineering *Ian Sommerville., Addison Wesley, 2010* • Yazılım Mühendisliği *Erhan Sarıdoğan, 1st ed., Papatya Yayıncılık, 2004*
Journals	• IEEE Transactions on Software Engineering • IEEE Software • ACM Transactions on Software Engineering and Methodology *Please click here for a larger list of journals.*
Societies	• The IEEE Computer Society
Other Links	• The Software Engineering Institute

Course - Outline

Week	Date	Topic	Homeworks/Projects
1	20.09	Introduction – Software Projects and Scope (Ch.1)	
2	27.09	Software Processes Models (Ch.2)	
3	04.10	Agile Software Development	Project Charter
4	11.10	Software Project Planning and Estimation I (Ch.9)	
5	18.10	Software Project Planning and Estimation II (Ch.9)	
6	25.10	Requirements Engineering (Ch.11) Recitation: JIRA	Project Plan
7	01.11	Requirements Analysis (Ch.12 - Ch.13) Recitation: BootStrap	
8	08.11	HOLIDAY	
9	15.11	Midterm Examination	Requirements Spec.
10	22.11	Software Architectures Recitation: ArgoUML	
11	29.11	Software Design I (Ch.14-17) Recitation: Hibernate, Log4J	
12	06.12	Software Design II (Ch.14-17)	Design Document
13	13.12	Implementation and Testing I(Ch.14-15)	
14	20.12	Implementation and Testing II(Ch.14-15) Recitation: Junit, Mockito, JMeter	
15	27.12	Project Presentations	Test Report

Course - Grading

Midterm Exam (x 1)	%25
Final Exam	%40
Project	%35
Total	%100

Tentative (subject to change)

1. Course Objectives
2. Software Projects
   a. Stakeholders
   b. Phases
3. Object Oriented Paradigm
4. Project Scope – How to begin a project?

Introduction

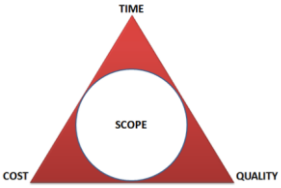
1.1

What is Software Engineering?

- Formal Definition
  - The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software"
    [IEEE Standard, 610.12, 1990].

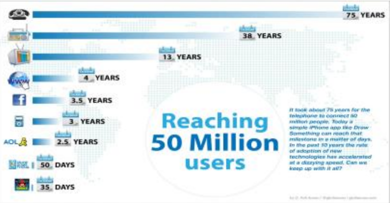
What is Software Engineering?

- The study of systematic and effective processes and technologies for supporting software development and maintenance activities
  - Improve quality
  - Reduce costs
  - Deliver on-time



What's the problem?

- Software cannot be built fast enough to keep up with technology
- Increasing need for high reliability software
- Software is difficult to maintain
- Difficult to estimate software costs and schedules
- Too many projects fail



Software Disaster Examples - 1

Therac-25 (1985)

- **Cost:** Three people dead, three people critically injured
- **Disaster:** Canada's Therac-25 radiation therapy machine malfunctioned and delivered lethal radiation doses to patients.
- **Cause:** Because of a subtle bug called a race condition, a technician could accidentally configure Therac-25 so the electron beam would fire in high-power mode without the proper patient shielding.

Software Disaster Examples -2

Patriot Missile (1991)

- **Cost:** 28 soldiers dead, 100 injured
- **Disaster:** During the first Gulf War, an American Patriot Missile system in Saudi Arabia failed to intercept an incoming Iraqi Scud missile. The missile destroyed an American Army barracks.
- **Cause:** A software rounding error incorrectly calculated the time, causing the Patriot system to ignore the incoming Scud missile.

Software Disaster Examples - 3

Ariane 5 Rocket (1996)

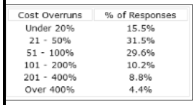
- **Cost:** $500 million
- **Disaster:** Ariane 5, Europe's newest unmanned rocket, was intentionally destroyed seconds after launch on its first flight. Also destroyed was its cargo of four scientific satellites to study how the Earth's magnetic field interacts with solar winds.
- **Cause:** Shutdown occurred when the guidance computer tried to convert the sideways rocket velocity from 64-bits to a 16-bit format. The number was too big, and an overflow error resulted. When the guidance system shut down, control passed to an identical redundant unit, which also failed because it was running the same algorithm.

Standish Project Benchmarks over the years

Year	Successful (%)	Challenged (%)	Failed (%)
1994	16	53	31
1996	27	33	40
1998	26	46	28
2000	28	49	23
2004	29	53	18
2006	35	46	19
2009	32	44	24

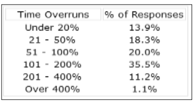
Overrunning??

Cost Overrun Data



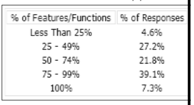
Cost Overruns	% of Responses
Under 20%	15.5%
21 - 50%	31.5%
51 - 100%	29.6%
101 - 200%	10.2%
201 - 400%	8.8%
Over 400%	4.4%

Time Overrun Data



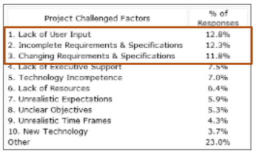
Time Overruns	% of Responses
Under 20%	13.9%
21 - 50%	18.3%
51 - 100%	20.0%
101 - 200%	35.5%
201 - 400%	11.2%
Over 400%	1.1%

# of Feature Dropped



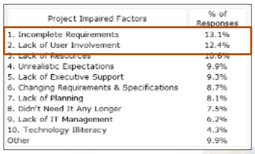
% of Features/Functions	% of Responses
Less Than 25%	4.6%
25 - 49%	27.2%
50 - 74%	21.8%
75 - 99%	39.1%
100%	7.3%

Factors Making SD Difficult



Project Challenged Factors	% of Responses
1. Lack of User Input	12.8%
2. Incomplete Requirements & Specifications	12.3%
3. Changing Requirements & Specifications	11.8%
4. Lack of Executive Support	7.5%
5. Technology Incompetence	7.0%
6. Lack of Resources	6.4%
7. Unrealistic Expectations	5.9%
8. Unclear Objectives	5.3%
9. Unrealistic Time Frames	4.3%
10. New Technology	3.7%
Other	23.0%

Factors Making SD Fail



Project Impaired Factors	% of Responses
1. Incomplete Requirements	13.1%
2. Lack of User Involvement	12.4%
3. Lack of Resources	10.6%
4. Unrealistic Expectations	9.9%
5. Lack of Executive Support	9.3%
6. Changing Requirements & Specifications	8.7%
7. Lack of Planning	8.1%
8. Didn't Need It Any Longer	7.5%
9. Lack of IT Management	6.2%
10. Technology Illiteracy	4.3%
Other	9.9%

Source : Standish Group

Why is software development so difficult?

- Communication
  - Between customer and developer
  - Within development team
- Project characteristics
  - Advancing technology
  - Changing requirements
- Personnel characteristics
  - Personnel variability
  - High turnover
- Facilities and resources
- Management issues

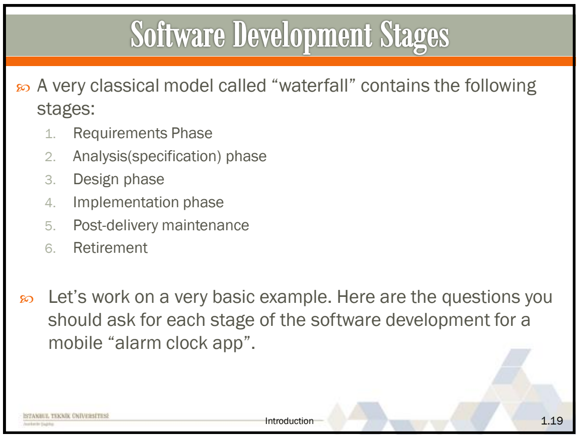
1. Course Objectives
2. Software Projects ⇦
   a. Phases
   b. Stakeholders
3. Object Oriented Paradigm
4. Project Scope – How to begin a project?

Software Projects

1.2

Software Business

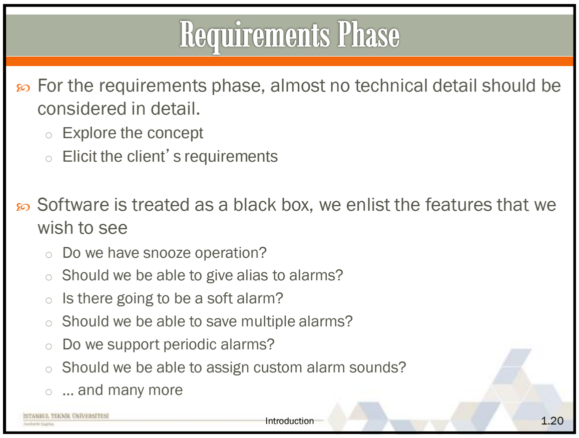
- Today's software development activities heavily rely on a project based approach.
- A project can be seen as a series of planned activities packed within a scope. The scope of a software project consists of
  - **Time:** How much time do we need to complete the project.
  - **Cost:** How much effort needed to complete the project on time.
  - **Quality:** What primary and secondary features and functionalities should be present regarding the time and budget.
- Software projects are generally carried out in phases (or stages) containing activities with the intent of better planning and management.



Software Development Stages

- A very classical model called “waterfall” contains the following stages:
  1. Requirements Phase
  2. Analysis(specification) phase
  3. Design phase
  4. Implementation phase
  5. Post-delivery maintenance
  6. Retirement

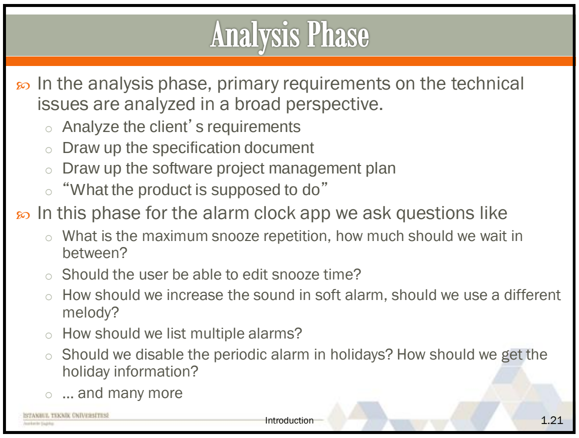
- Let’s work on a very basic example. Here are the questions you should ask for each stage of the software development for a mobile “alarm clock app”.



Requirements Phase

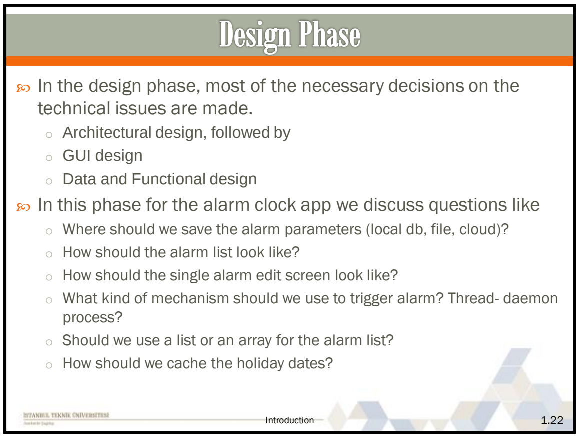
- For the requirements phase, almost no technical detail should be considered in detail.
  - Explore the concept
  - Elicit the client’ s requirements

- Software is treated as a black box, we enlist the features that we wish to see
  - Do we have snooze operation?
  - Should we be able to give alias to alarms?
  - Is there going to be a soft alarm?
  - Should we be able to save multiple alarms?
  - Do we support periodic alarms?
  - Should we be able to assign custom alarm sounds?
  - … and many more



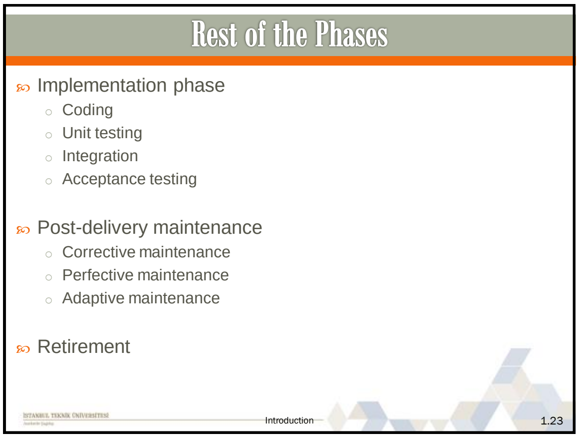
Analysis Phase

- In the analysis phase, primary requirements on the technical issues are analyzed in a broad perspective.
  - Analyze the client’ s requirements
  - Draw up the specification document
  - Draw up the software project management plan
  - “What the product is supposed to do”
- In this phase for the alarm clock app we ask questions like
  - What is the maximum snooze repetition, how much should we wait in between?
  - Should the user be able to edit snooze time?
  - How should we increase the sound in soft alarm, should we use a different melody?
  - How should we list multiple alarms?
  - Should we disable the periodic alarm in holidays? How should we get the holiday information?
  - … and many more



Design Phase

- In the design phase, most of the necessary decisions on the technical issues are made.
  - Architectural design, followed by
  - GUI design
  - Data and Functional design
- In this phase for the alarm clock app we discuss questions like
  - Where should we save the alarm parameters (local db, file, cloud)?
  - How should the alarm list look like?
  - How should the single alarm edit screen look like?
  - What kind of mechanism should we use to trigger alarm? Thread- daemon process?
  - Should we use a list or an array for the alarm list?
  - How should we cache the holiday dates?

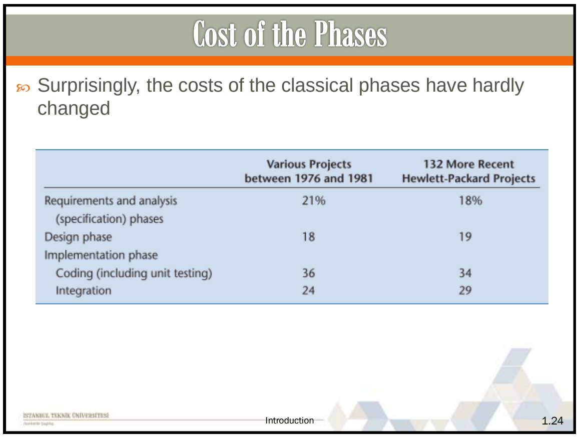


Rest of the Phases

- Implementation phase
  - Coding
  - Unit testing
  - Integration
  - Acceptance testing

- Post-delivery maintenance
  - Corrective maintenance
  - Perfective maintenance
  - Adaptive maintenance

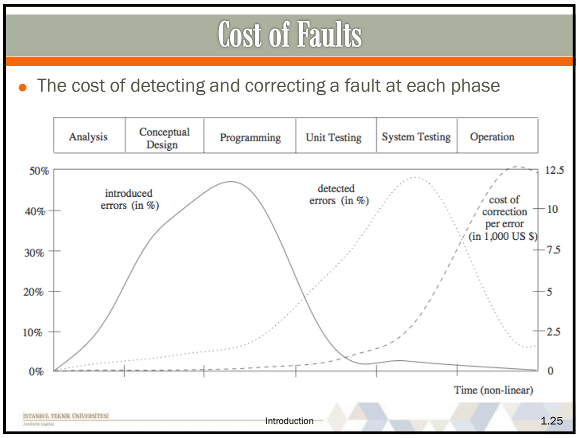
- Retirement



Cost of the Phases

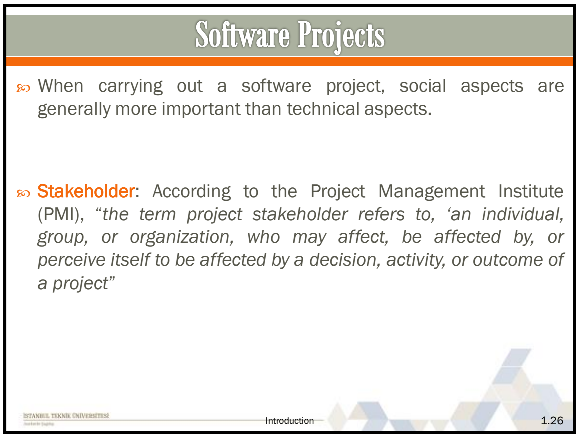
- Surprisingly, the costs of the classical phases have hardly changed

	Various Projects between 1976 and 1981	132 More Recent Hewlett-Packard Projects
Requirements and analysis (specification) phases	21%	18%
Design phase	18	19
Implementation phase		
Coding (including unit testing)	36	34
Integration	24	29



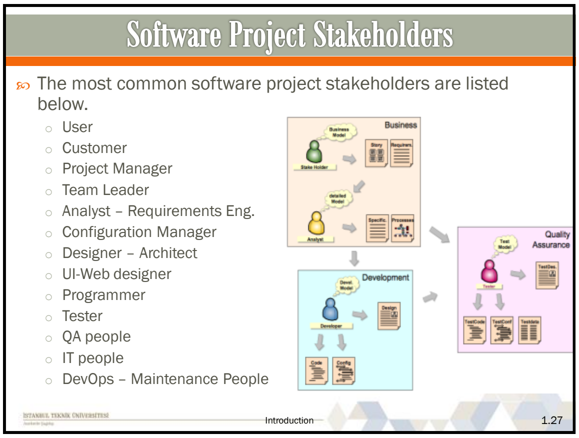
Cost of Faults

- The cost of detecting and correcting a fault at each phase



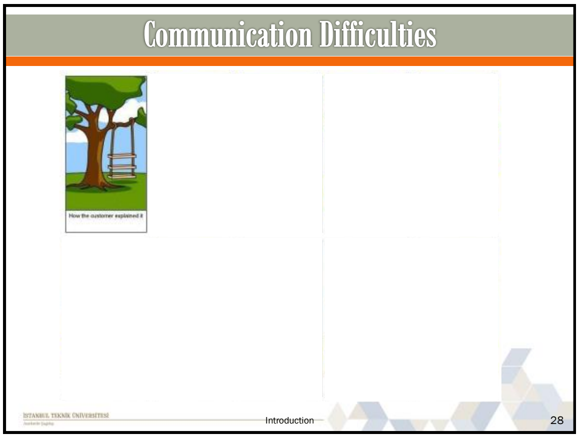
Software Projects

- When carrying out a software project, social aspects are generally more important than technical aspects.
- Stakeholder: According to the Project Management Institute (PMI), *"the term project stakeholder refers to, 'an individual, group, or organization, who may affect, be affected by, or perceive itself to be affected by a decision, activity, or outcome of a project"*

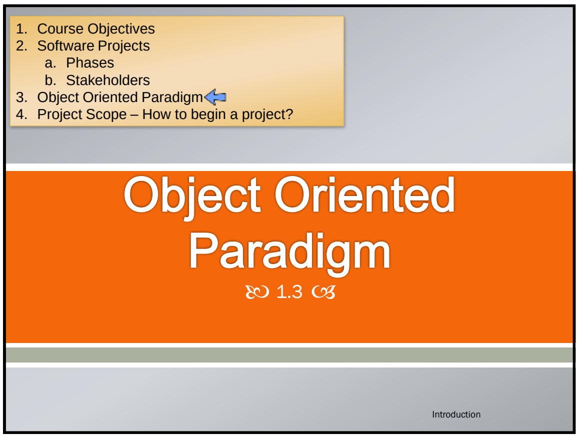


Software Project Stakeholders

- The most common software project stakeholders are listed below.
  - User
  - Customer
  - Project Manager
  - Team Leader
  - Analyst – Requirements Eng.
  - Configuration Manager
  - Designer – Architect
  - UI-Web designer
  - Programmer
  - Tester
  - QA people
  - IT people
  - DevOps – Maintenance People



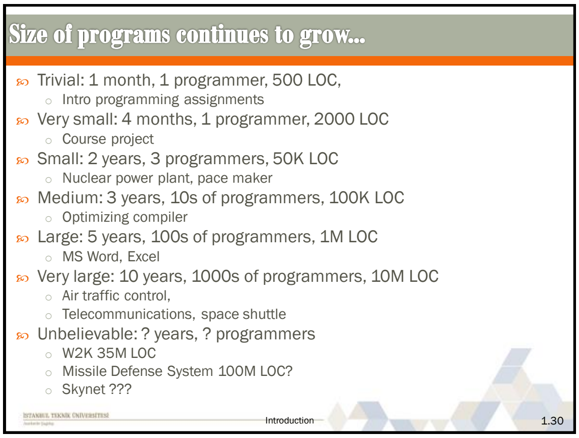
Communication Difficulties



1. Course Objectives
2. Software Projects
   a. Phases
   b. Stakeholders
3. Object Oriented Paradigm
4. Project Scope – How to begin a project?

Object Oriented Paradigm

1.3



Size of programs continues to grow...

- Trivial: 1 month, 1 programmer, 500 LOC,
  - Intro programming assignments
- Very small: 4 months, 1 programmer, 2000 LOC
  - Course project
- Small: 2 years, 3 programmers, 50K LOC
  - Nuclear power plant, pace maker
- Medium: 3 years, 10s of programmers, 100K LOC
  - Optimizing compiler
- Large: 5 years, 100s of programmers, 1M LOC
  - MS Word, Excel
- Very large: 10 years, 1000s of programmers, 10M LOC
  - Air traffic control,
  - Telecommunications, space shuttle
- Unbelievable: ? years, ? programmers
  - W2K 35M LOC
  - Missile Defense System 100M LOC?
  - Skynet ???

The Object-Oriented Paradigm

- The structured paradigm was successful initially
  - It started to fail with larger products (> 50,000 LOC)
- Post-delivery maintenance problems (today, 70 to 80% of total effort)
- Reason: Structured methods are
  - Action oriented (e.g., finite state machines, data flow diagrams); or
  - Data oriented (e.g., entity-relationship diagrams, Jackson's method);
  - But not both

The Object-Oriented Paradigm

- Both data and actions are of equal importance
- Object:
  - A software component that incorporates both data and the actions that are performed on that data
- Example:
  - Bank account
    - Data: account balance
    - Actions: deposit, withdraw, determine balance

Structured vs Object-Oriented

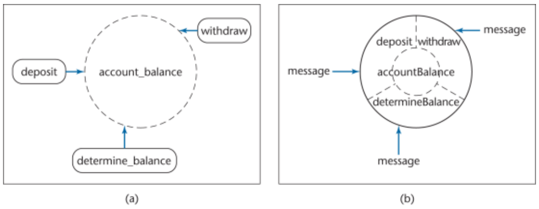


Figure 1.7

- Information hiding
- Responsibility-driven design
- Impact on maintenance, development

Information Hiding

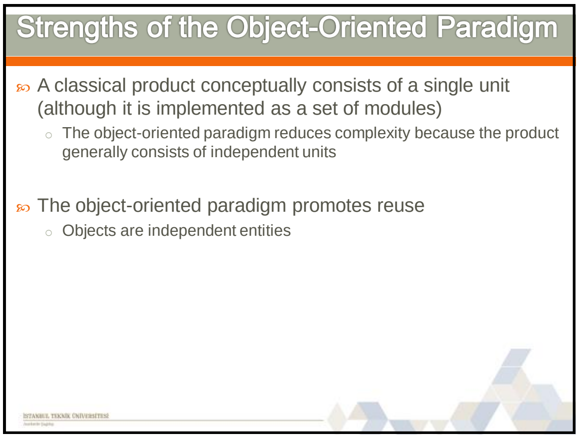
- In the object-oriented version
  - The solid line around `accountBalance` denotes that outside the object there is no knowledge of how `accountBalance` is implemented
- In the classical version
  - All the modules have details of the implementation of `account_balance`

Strengths of the OO Paradigm

- With information hiding, postdelivery maintenance is safer
  - The chances of a regression fault are reduced
- Development is easier
  - Objects generally have physical counterparts
  - This simplifies modeling (a key aspect of the object-oriented paradigm)

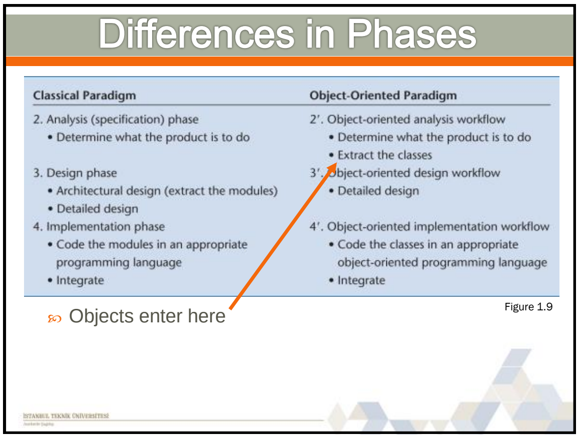
Strengths of the Object-Oriented Paradigm

- Well-designed objects are independent units
  - Everything that relates to the real-world item being modeled is in the corresponding object — *encapsulation*
  - Communication is by sending *messages*
  - This independence is enhanced by *responsibility-driven design*
- Send flowers to your mother in Chicago
  - Call `1-800-flowers`
  - Where is `1-800-flowers`?
  - Which Chicago florist does the delivery?
  - Information hiding
  - Send a message to a method [action] of an object without knowing the internal structure of the object



Strengths of the Object-Oriented Paradigm

- A classical product conceptually consists of a single unit (although it is implemented as a set of modules)
  - The object-oriented paradigm reduces complexity because the product generally consists of independent units
- The object-oriented paradigm promotes reuse
  - Objects are independent entities

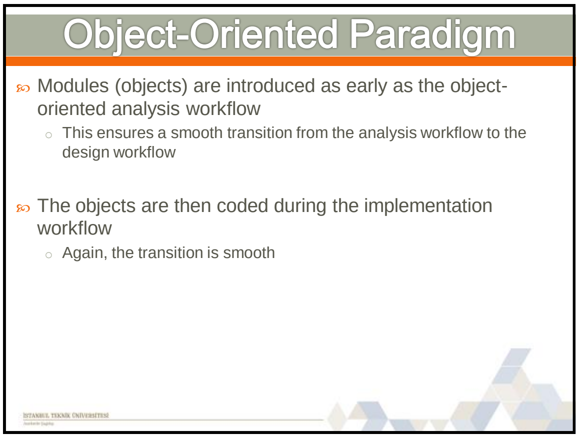


Differences in Phases

Classical Paradigm	Object-Oriented Paradigm
2. Analysis (specification) phase	2′. Object-oriented analysis workflow
• Determine what the product is to do	• Determine what the product is to do
	• Extract the classes
3. Design phase	3′. Object-oriented design workflow
• Architectural design (extract the modules)	• Detailed design
• Detailed design	
4. Implementation phase	4′. Object-oriented implementation workflow
• Code the modules in an appropriate programming language	• Code the classes in an appropriate object-oriented programming language
• Integrate	• Integrate

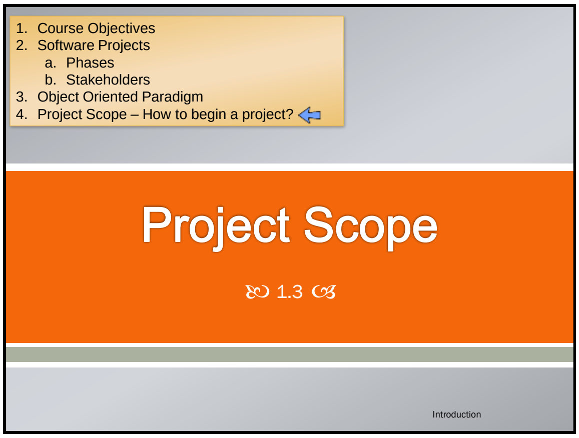
Figure 1.9

- Objects enter here



Object-Oriented Paradigm

- Modules (objects) are introduced as early as the object-oriented analysis workflow
  - This ensures a smooth transition from the analysis workflow to the design workflow
- The objects are then coded during the implementation workflow
  - Again, the transition is smooth

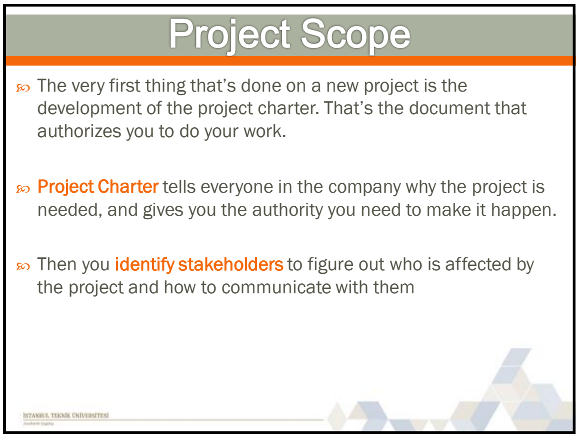


1. Course Objectives
2. Software Projects
   a. Phases
   b. Stakeholders
3. Object Oriented Paradigm
4. Project Scope – How to begin a project?

Project Scope

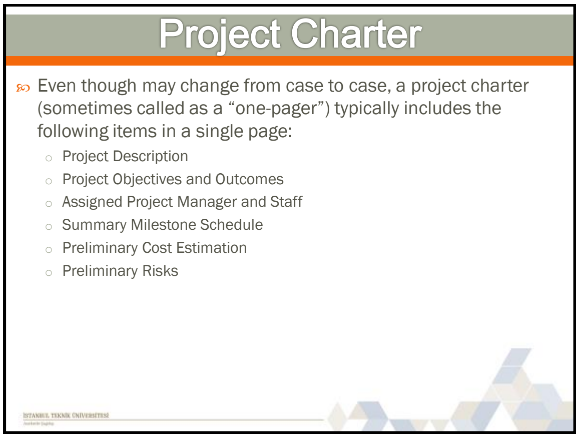
1.3

Introduction



Project Scope

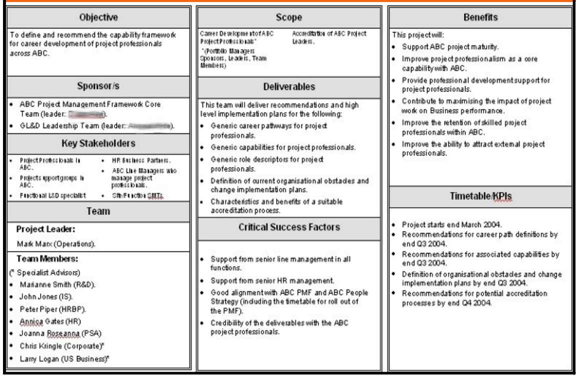
- The very first thing that's done on a new project is the development of the project charter. That's the document that authorizes you to do your work.
- Project Charter tells everyone in the company why the project is needed, and gives you the authority you need to make it happen.
- Then you identify stakeholders to figure out who is affected by the project and how to communicate with them



Project Charter

- Even though may change from case to case, a project charter (sometimes called as a "one-pager") typically includes the following items in a single page:
  - Project Description
  - Project Objectives and Outcomes
  - Assigned Project Manager and Staff
  - Summary Milestone Schedule
  - Preliminary Cost Estimation
  - Preliminary Risks

Project Charter



Objective

To define and recommend the capability framework for career development of project professionals across ABC.

Sponsor/s

- ABC Project Management Framework Core Team (leader: [redacted]).
- GL&D Leadership Team (leader: [redacted]).

Key Stakeholders

- Project Professionals in ABC.
- Projects upportgroups in ABC.
- Functional L&D specialist
- HR Business Partners.
- ABC Line Managers who manage project professionals.
- Site/Function SMTs.

Team

Project Leader:

Mark Marx (Operations).

Team Members:

(* Specialist Advisors)

- Marianne Smith (R&D).
- John Jones (IS).
- Peter Piper (HRBP).
- Annica Gates (HR)
- Joanna Roseanna (PSA)
- Chris Kringle (Corporate)*
- Larry Logan (US Business)*

Scope

Career Development of ABC Project Professionals*

*(Portfolio Managers Sponsors, Leaders, Team Members)

Accreditation of ABC Project Leaders.

Deliverables

This team will deliver recommendations and high level implementation plans for the following:

- Generic career pathways for project professionals.
- Generic capabilities for project professionals.
- Generic role descriptors for project professionals.
- Definition of current organisational obstacles and change implementation plans.
- Characteristics and benefits of a suitable accreditation process.

Critical Success Factors

- Support from senior line management in all functions.
- Support from senior HR management.
- Good alignment with ABC PMF and ABC People Strategy (including the timetable for roll out of the PMF).
- Credibility of the deliverables with the ABC project professionals.

Benefits

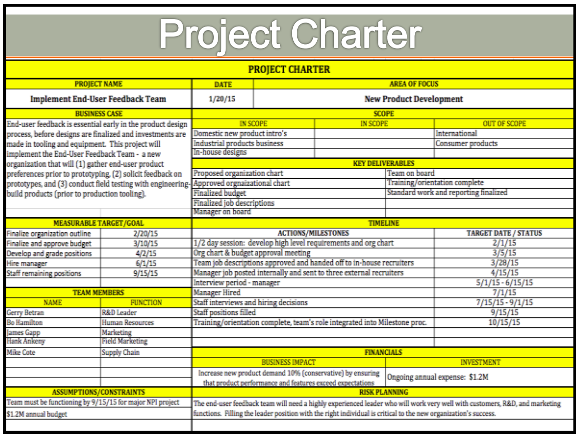
This project will:

- Support ABC project maturity.
- Improve project professionalism as a core capability with ABC.
- Provide professional development support for project professionals.
- Contribute to maximising the impact of project work on Business performance.
- Improve the retention of skilled project professionals within ABC.
- Improve the ability to attract external project professionals.

Timetable/KPIs

- Project starts end March 2004.
- Recommendations for career path definitions by end Q3 2004.
- Recommendations for associated capabilities by end Q3 2004.
- Definition of organisational obstacles and change implementation plans by end Q3 2004.
- Recommendations for potential accreditation processes by end Q4 2004.

Project Charter



PROJECT CHARTER

PROJECT NAME	DATE	AREA OF FOCUS
Implement End-User Feedback Team	1/20/15	New Product Development

BUSINESS CASE

End-user feedback is essential early in the product design process, before designs are finalized and investments are made in tooling and equipment. This project will implement the End-User Feedback Team - a new organization that will (1) gather end-user product preferences prior to prototyping, (2) solicit feedback on prototypes, and (3) conduct field testing with engineering-build products (prior to production tooling).

SCOPE

IN SCOPE	IN SCOPE	OUT OF SCOPE
Domestic new product intro's		International
Industrial products business		Consumer products
In-house designs		

KEY DELIVERABLES

Proposed organization chart	Team on board
Approved orgnaizational chart	Training/orientation complete
Finalized budget	Standard work and reporting finalized
Finalized job descriptions	
Manager on board	

MEASURABLE TARGET/GOAL

Finalize organization outline	2/20/15
Finalize and approve budget	3/10/15
Develop and grade positions	4/2/15
Hire manager	6/1/15
Staff remaining positions	9/15/15

TIMELINE

ACTIONS/MILESTONES	TARGET DATE / STATUS
1/2 day session: develop high level requirements and org chart	2/1/15
Org chart & budget approval meeting	3/5/15
Team job descriptions approved and handed off to in-house recruiters	3/28/15
Manager job posted internally and sent to three external recruiters	4/15/15
Interview period - manager	5/1/15 - 6/15/15
Manager Hired	7/1/15
Staff interviews and hiring decisions	7/15/15 - 9/1/15
Staff positions filled	9/15/15
Training/orientation complete, team's role integrated into Milestone proc.	10/15/15

TEAM MEMBERS

NAME	FUNCTION
Gerry Betran	R&D Leader
Bo Hamilton	Human Resources
James Gapp	Marketing
Hank Ankeny	Field Marketing
Mike Cote	Supply Chain

FINANCIALS

BUSINESS IMPACT	INVESTMENT
Increase new product demand 10% (conservative) by ensuring that product performance and features exceed expectations	Ongoing annual expense: $1.2M

ASSUMPTIONS/CONSTRAINTS

Team must be functioning by 9/15/15 for major NPI project

$1.2M annual budget

RISK PLANNING

The end-user feedback team will need a highly experienced leader who will work very well with customers, R&D, and marketing functions. Filling the leader position with the right individual is critical to the new organization's success.

Project Scope

- Once you have a good idea of what needs to be done, you need to **track your scope** as the project work is happening. Determining the project scope is setting goals for the project team and keep everybody on track.
  - Product scope means the features and functions of the product or service that you and your team are building.
  - Project scope is all of the work that needs to be done to make the product.
  - Scope creep means uncontrolled changes that cause the team to do extra work.

Project Scope

- The five Scope Management processes that can be used in scope management are
  - Collecting the requirements to form a requirements document
  - Defining the Scope to form a Project Scope document
  - Creating a work breakdown structure
  - Consider change requests to modify project scope
  - Verify the scope iteratively by accepted deliverables

Project Scope Document

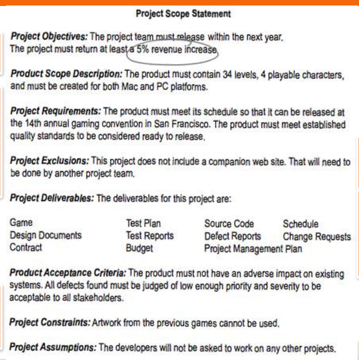
- The Project Scope document is created by considering the following documents:
  - Project Charter
  - Requirements Document
  - Organizational Templates/Forms
- When creating the project scope statement, you can perform the following actions
  - Stakeholder meetings → Output: Quantifiable goals
  - Product analysis
  - Alternatives Identification
  - Expert Judgement

Project Scope Document

All the project objectives need to be measureable

Even though you probably can't fit ALL of the requirements here, there should be enough detail to let you keep on planning and refer back to it late

Constraints are known limitations. Assumptions are things you think are true.



Project Scope Statement

Project Objectives: The project team must release within the next year. The project must return at least a 5% revenue increase

Product Scope Description: The product must contain 34 levels, 4 playable characters, and must be created for both Mac and PC platforms.

Project Requirements: The product must meet its schedule so that it can be released at the 14th annual gaming convention in San Francisco. The product must meet established quality standards to be considered ready to release.

Project Exclusions: This project does not include a companion web site. That will need to be done by another project team.

Project Deliverables: The deliverables for this project are:

Game	Test Plan	Source Code	Schedule
Design Documents	Test Reports	Defect Reports	Change Requests
Contract	Budget	Project Management Plan	

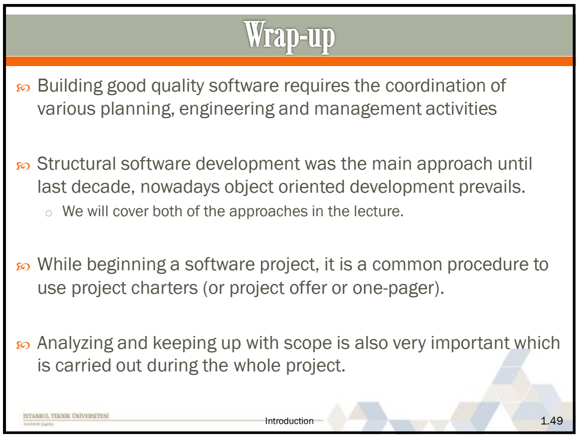
Product Acceptance Criteria: The product must not have an adverse impact on existing systems. All defects found must be judged of low enough priority and severity to be acceptable to all stakeholders.

Project Constraints: Artwork from the previous games cannot be used.

Project Assumptions: The developers will not be asked to work on any other projects.

This means looking for all the work the project DOESN'T include.

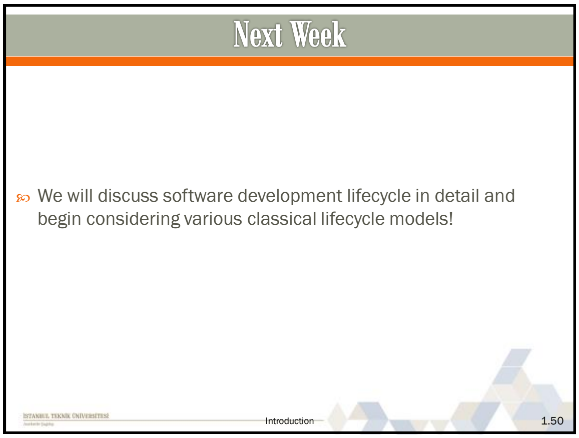
The deliverables listed here are EVERYTHING the project creates, including project management stuff.



Wrap-up

- Building good quality software requires the coordination of various planning, engineering and management activities
- Structural software development was the main approach until last decade, nowadays object oriented development prevails.
  - We will cover both of the approaches in the lecture.
- While beginning a software project, it is a common procedure to use project charters (or project offer or one-pager).
- Analyzing and keeping up with scope is also very important which is carried out during the whole project.

İSTANBUL TEKNİK ÜNİVERSİTESİ Introduction 1.49



Next Week

- We will discuss software development lifecycle in detail and begin considering various classical lifecycle models!

İSTANBUL TEKNİK ÜNİVERSİTESİ Introduction 1.50