HY-351:
Ανάλυση και Σχεδίαση Πληροφοριακών Συστημάτων
Information Systems Analysis and Design

Πανεπιστήμιο Κρήτης, Φθινόπωρο 2005

Γιάννης Τζίτζικας

Διάλεξη : 1
Ημερομηνία : 27-9-2005
Θέμα : Διαδικαστικά και Εισαγωγή

HY351: Στοιχεία Μαθήματος

- Διδακτικές μονάδες: 4
- Προαπαιτούμενα
  - Αντικειμενοστραφής Προγραμματισμός (HY 252)
- Συνιστώμενα
  - Αρχεία και Βάσεις Δεδομένων (HY 360)
  - Τεχνολογία Λογισμικού (HY 352)
- Εβδομαδιαίο Πρόγραμμα :
  - **Διαλέξεις**: Τρίτη 3-5 και Πέμπτη 3-5 (αίθουσα PA201)
  - **Φροντιστήρια**: Δευτέρα 3-5 *ή 7-9* (αίθουσα Λ202)
- Παρακολούθηση
  - Αναμενόμενη αλλά όχι υποχρεωτική
- Γραφτείτε (από αύριο) στη λίστα **`hy351-list`**

Προσωπικό και Ιστοσελίδα Μαθήματος

- Διδάσκων:
  - Γιάννης Τζίτζικας
  - Ώρες γραφείου (Γ111) : κυρίως μετά τις διαλέξεις ή ύστερα από συνεννόηση
- Βοηθοί:
  - Κώστας Βανδίκας
  - Νίκος Δημαρέσης
  - Γιάννης Καπανταιδάκης
- www.csd.uoc.gr/~hy351
  - Τελευταίες Ανακοινώσεις
  - Περιγραφή Μαθήματος - Διδακτέα Ύλη
  - Πρόγραμμα Διαλέξεων
  - Διαφάνειες Διαλέξεων, Πρόγραμμα Μελέτης
  - Ασκήσεις, Λύσεις, Βαθμολογίες
  - Βιβλιογραφία
  - Σύνδεσμοι προς συμπληρωματικό διδακτικό υλικό (βιβλία, άρθρα, σχετικές διαδυκτιακές πύλες κλπ).

Συνοπτική Περιγραφή Μαθήματος

Τα σύγχρονα πληροφοριακά συστήματα καλύπτουν ένα ευρύτατο φάσμα εφαρμογών, από την διεκπεραίωση πολύπλοκων επιχειρησιακών λειτουργιών, την συσσώρευση επιχειρησιακής γνώσης και την στήριξη διαδικασιών λήψης αποφάσεων, μέχρι την δημιουργία συστημάτων τεκμηρίωσης και την παροχή εξατομικευμένων υπηρεσιών πληροφόρησης. Το τεχνολογικό υπόβαθρο των πληροφοριακών συστημάτων περιλαμβάνει συστήματα βάσεων δεδομένων, repositories, data warehouses, συστήματα ανάκτησης πληροφοριών και τηλεπικοινωνιακές τεχνολογίες. Για την σχεδίαση και την κατασκευή των συστημάτων αυτών έχουν αναπτυχθεί και εξακολουθούν να αναπτύσσονται ειδικές συστηματικές μέθοδοι. Το μάθημα προσφέρει μια συστηματική εισαγωγή στην ανάλυση και σχεδίαση πληροφοριακών συστημάτων και καλύπτει θεωρητικά, τεχνικά και μεθοδολογικά ζητήματα. Το μάθημα θα επιτρέψει στους φοιτητές να εξοικειωθούν με τις πιο σημαντικές έννοιες, αρχές, και στάδια ανάλυσης και σχεδίασης πληροφοριακών συστημάτων.

Στόχοι του Μαθήματος

Με την συμπλήρωση αυτού του μαθήματος, κάθε φοιτητής πρέπει να:

- έχει κατανοήσει το ρόλο της ανάλυσης και της σχεδίασης πληροφοριακών συστημάτων
- να έχει κατανοήσει τους τρόπους προγραμματισμού και διοίκησης ενός έργου
- έχει εξοικειωθεί με τη συλλογή και οργάνωση πληροφοριών για έναν οργανισμό και να ξέρει να συντάσσει μια **μελέτη σκοπιμότητας** για ένα πλ. σύστημα
- μπορεί να προδιαγράφει τις **λειτουργικές και μη λειτουργικές απαιτήσεις** ενός συστήματος καθώς και τον τρόπο χρήσης του με Περιπτώσεις Χρήσης
- έχει μάθει πώς να μοντελοποιεί τις διάφορες απόψεις ενός συστήματος (δομή, συμπεριφορά, αλληλεπίδραση, καταστάσεις, περιορισμούς, αρχιτεκτονική, κ.α.).
- μπορεί να σχεδιάζει την **αρχιτεκτονική** ενός πλ. συστήματος
- μπορεί να συντάσσει πλουσιότερες περιγραφές ενός σχεδίου χρησιμοποιώντας **UML** διαγράμματα (διαγράμματα κλάσεων, καταστάσεων, εργασιών, αλληλεπίδρασης)
- μπορεί να σχεδιάσει την **Βάση Δεδομένων** και την **Επαφή Χρήσης** ενός πλ. συστήματος
- έχει κατανοήσει τεχνικές που μπορούν να αυξήσουν την ευελιξία ενός σχεδίου και άλλες αρχές καλής σχεδίασης

Στόχοι του Μαθήματος (ΙΙ)

Θα δοθεί έμφαση στην Αντικειμενοστρεφή Ανάλυση και Σχεδίαση Πληροφοριακών Συστημάτων

Θα δοθεί έμφαση στη χρήση των εργαλείων CASE ως ουσιαστική βοήθεια για την ανάλυση και την σχεδίαση συστημάτων, και ειδικότερα την χρήση της ενοποιημένης γλώσσας μοντελοποίησης UML.

Το μάθημα βασίζεται σε ευρέως αποδεκτές πρακτικές που έχουν αποδειχθεί ότι βελτιώνουν την ποιότητα ενός πληροφοριακού συστήματος ενώ παράλληλα μειώνουν τον χρόνο ανάπτυξης και συντήρησής του.

Θεματικές Ενότητες

[Α] Εισαγωγή

[Β] Φάση Προγραμματισμού (planning)

[Γ] Φάση Ανάλυσης

[Δ] Φάση Σχεδιασμού

[Ε] Φάση Υλοποίησης

Θεματικές Ενότητες
(Α) Εισαγωγή

1. Εισαγωγή στην Ανάλυση και Σχεδίαση Πλ. Συστημάτων

- Τύποι Πληροφοριακών Συστημάτων
- Ο κύκλος ζωής ενός Π.Σ.
- Μεθοδολογίες ανάπτυξης

2. Εισαγωγή στην Αντικειμενοστρεφή Ανάλυση και Σχεδίαση με τη χρήση της UML

- Βασικές αρχές αντικειμενοστρεφούς σχεδιασμού
- Εισαγωγή και Περιληπτική Σύνοψη της UML
- Χρήση της UML στην Ανάλυση και Σχεδίαση

Θεματικές Ενότητες
(Β) Φάση Προγραμματισμού (planning)

3. Έναρξη Έργου

·Εντοπισμός Προβλήματος και Προσδιορισμός εμβέλειας έργου

·Μελέτη Επιτευξιμότητας (τεχνικής, οικονομικής, επιχειρησιακής)

4. Διοίκηση Έργου

·Εκτίμηση μεγέθους έργου

·Σύνταξη πλάνου εργασίας (workplan)

·Διαγράμματα Gantt και PERT

·Έλεγχος και συντονισμός έργου

Θεματικές Ενότητες
(Γ) Φάση Ανάλυσης

5. Καθορισμός Απαιτήσεων (Requirements Determination)

·Τεχνικές συλλογής, ανάλυσης και οργάνωσης απαιτήσεων

·Λειτουργικές και Μη-Λειτουργικές απαιτήσεις

6. Μοντελοποίηση Λειτουργιών (Functional Modeling)

·Περιπτώσεις Χρήσης (Use Cases)

·Διαγράμματα Δραστηριοτήτων (Activity Diagrams)

7. Μοντελοποίηση Δομής (Structural Modeling)

·Διαγράμματα Κλάσεων, CRC Cards

8. Μοντελοποίηση Συμπεριφοράς (Behavioral Modeling)

·Διαγράμματα Αλληλεπίδρασης (Interaction Diagrams) και Καταστάσεων

9. Έκφραση Περιορισμών με χρήση της OCL

Θεματικές Ενότητες
(Δ) Φάση Σχεδίασης

10. Από την ανάλυση στη σχεδίαση
- · Διαγράμματα Συσκευασίας (Package diagrams)
- · Στρατηγικές Σχεδίασης
- . Σχεδίαση Κλάσεων και Μεθόδων

11. Σχεδίαση Διαχείρισης Δεδομένων
- ·Το μοντέλο Οντοτήτων-Συσχετίσεων (ER)
- ·Σχεδίαση Σχεσιακής Βάσης Δεδομένων

12. Σχεδίαση Αλληλεπίδρασης Ανθρώπου Μηχανής
- ·Αρχές, διαδικασία σχεδίασης αλληλεπίδρασης
- ·Σχεδίαση εισαγωγής και εξαγωγής δεδομένων

13. Σχεδίαση Φυσικής Αρχιτεκτονικής
- ·Στοιχεία της Φυσικής Αρχιτεκτονικής
- ·Μη-λειτουργικές απαιτήσεις και φυσική αρχιτεκτονική
- ·Επιλογή υλικού, λογισμικού και αρχιτεκτονικής λογισμικού
- ·Διαγράμματα Εξαρτημάτων (component) και Παράταξης (deployment)

Θεματικές Ενότητες
(Ε) Φάση Υλοποίησης

14. Κατασκευή
- ·Κατανομή και συντονισμός προγραμματιστικού έργου
- ·Σχεδιασμός Testing
- ·Τεκμηρίωση (documentation)

15. Εγκατάσταση και Συντήρηση
- ·Μετάβαση
- ·Διαχείριση Αλλαγών
- ·Άλλες δραστηριότητας μετά την εγκατάσταση

16. Άλλα ζητήματα
- ·Δοκιμές και Ενοποίηση Μονάδων
- ·Αντίστροφη Μηχανολογία (reverse engineering) Πληροφοριακών Συστημάτων
- ·Μελέτες Περιπτώσεων

Βιβλιογραφία

- **UML**
  - **UML Distilled: A Brief Guide to the Standard Object Modeling Language** (3rd Edition) by Martin Fowler, Addison Wesley, 2004.
  - **The Unified Modeling Language User Guide** (2nd edition) by G. Booch, J. Rumbaugh, I. Jacobson, Addison Wesley, 2004
- **IS Analysis and Design**
  - **Systems Analysis and Design with UML Version 2.0** (2nd edition) by A. Dennis, B. Haley Wixom, D. Tegarden, Wiley, 2005
  - **Requirements Analysis and System Design** (2nd edition) by Leszek A. Maciaszek, Addison Wesley, 2005
  - **System Analysis and Design Methods** (6th edition) by Jeffrey L. Whitten, Lonnie D. Bentley and Kevin Dittman, McGraw-Hill, 2004
  - **Object-Oriented Systems Analysis and Design Using UML** (2nd edition) by S. Bennett, S. McRobb, R. Farmer, McGraw Hill, 2002.
  - **Object Design: Roles, Responsibilities and Collaborations** by Rebecca Wirfs-Brock and Alan McKean, Addison-Wesley, 2003
  - **Modern Systems Analysis & Design** (4th Edition) by Jeffrey A. Hoffer, Joef F. George, Joseph S. Valacich, Prentice Hall, 2005

Ελληνική Βιβλιογραφία

- Shari Lawrence Pfleeger. Τεχνολογία Λογισμικού: Θεωρία και Πράξη, 1. Κλειδάριθμος, Αθήνα, 2003.
- Shari Lawrence Pfleeger. Τεχνολογία Λογισμικού: Θεωρία και Πράξη, 2. Κλειδάριθμος, Αθήνα, 2004.
- Ε. Κιουντούζης, Μεθοδολογίες Ανάλυσης και Σχεδιασμού Πληροφοριακών Συστημάτων,Εκδόσεις Α.Σταμούλη, Αθήνα 1997
- Β. Λαοπόδης, ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ,ΥΛΟΠΟΙΗΣΗ & ΜΑΝΑΤΖΜΕΝΤ, Κλειδάριθμος
- Β. Λαοπόδης, Ανάλυση και σχεδιασμός συστημάτων, Κλειδάριθμος 1996
- Αρ. Μακρής, ΣΧΕΔΙΑΣΜΟΣ ΠΛΗΡΟΦ.ΣΥΣΤΗΜΑΤΩΝ & ΣΧΕΣΙΑΚΩΝ ΒΔ,, Κλειδάριθμος 2002
- Εμμ. Α. Γιακουμάκης, Τεχνολογία Λογισμικού: Απαιτήσεις Λογισμικού, σχεδίαση λογισμικού, Εκδόσεις Α. Σταμούλης, Αθήνα, Πειραιάς, 1994.
- Εμμ. Α. Γιακουμάκης, Τεχνολογία Λογισμικού: Κωδικοποίηση, έλεγχος και συντήρηση λογισμικού, Εκδόσεις Α. Σταμούλης, Αθήνα, Πειραιάς, 1993
- Malaga Ross, Εισαγωγή στην Τεχνολογία Πληροφοριακών Συστημάτων, Γκιούρδας 2004
- …..

Βαθμολόγηση

- **Τελικός βαθμός**
  - **Βαθμός** = 40% **Έργο** + 20% **Ασκήσεις** + 40% **ΤελικήΕξέταση**
- Για να περάσετε το μάθημα χρειάζεστε
  - **Βαθμός** ≥ 5 **AND** ΤελικήΕξ ≥ 4
- Σημειώσεις στην Τελική Εξέταση:
  - Ανοιχτές

Εντιμότητα:

- Αντιγραφή ή άλλες μορφές κλοπής θα σημάνουν αποτυχία στο μάθημα
- Συμβουλές
  - μην αντιγράφετε ή δίνετε τις εργασίες σας σε άλλους
  - προστατέψτε τα αρχεία και τα έγγραφά σας
  - πάντα να αναφέρετε τις πηγές σας (άτομα, βιβλία, Web)

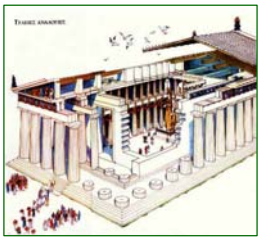
Έργο (Project)

- Ομάδες 2-3 ατόμων
- 3 φάσεις
  - 1η Φάση (Οκτώβριος): Μελέτη Σκοπιμότητας
  - 2η Φάση (Νοέμβριος): Ανάλυση Απαιτήσεων
  - 3η Φάση (Δεκέμβριος): Σχεδίαση Συστήματος

Σειρές Ασκήσεων

- Σκοπός: η εστίαση σε συγκεκριμένα ζητήματα

HY 351: Ανάλυση και Σχεδίαση Πληροφοριακών Συστημάτων
CS 351: Information Systems Analysis and Design



Ι. Εισαγωγή

Lecture : 1
Date : 27-9-2005

Yannis Tzitzikas
University of Crete, Fall 2005

Outline

- Types of Software
- What is an Information System
- Technological background of Information Systems
- Software Development: distinctive characteristics
- Software Engineering
- Why do Analysis and Design ?
- The triangle of success / failure
  - The stakeholders
  - The methodology
  - The tools

Τύποι Λογισμικού

- **Γενικές κατηγορίες**
  - Λογισμικό Συστημάτων (*Systems Software*)
  - Λογισμικό Πραγματικού Χρόνου (*Real-time Software*)
  - Πληροφοριακά Συστήματα (*Information Systems*)
  - Τεχνικές και Επιστημονικές Εφαρμογές (*Engineering and Scientific Applications*)
  - Ενσωματωμένο Λογισμικό (*Embedded Software*)
  - Λογισμικό Προσωπικού Υπολογιστή (*Personal Computer Software*)
  - Λογισμικό Ιστού (*Web Software*)
- **Το λογισμικό μπορεί να κατασκευάζεται για:**
  - Μία ευρεία κατηγορία πελατών-χρηστών (*generic software*)
  - Κατά παραγγελία για ένα συγκεκριμένο πελάτη-χρήστη
    - το μεγαλύτερο ποσοστό του λογισμικού κατασκευάζεται κατά παραγγελία.
- **Ανάλογα με τις δυνατότητες πρόσβασης στον πηγαίο κώδικα διακρίνουμε:**
  - Λογισμικό κλειστού κώδικα (*closed source software)*
  - Λογισμικό ανοιχτού κώδικα (*open source software*)

Πληροφοριακά Συστήματα (Information Systems)

- Είναι συνήθως κατά παραγγελία λογισμικό
- Αποτελούν την πληροφοριακή υποδομή της επιχείρησης
  - χρησιμοποιούνται ευρέως σε μεγάλες επιχειρήσεις
- Συχνά ενσωματώνουν διάφορα είδη λογισμικού
- Χρησιμοποιούνται τόσο στη διεκπεραίωση (*back office)* όσο και στις παραγωγικές διαδικασίες (*core business*).
  - decision support (DSS)
  - on-line analytical processing (OLAP)
  - data mining
  - for customer service (web-based systems)

Πληροφοριακά Συστήματα

- Τα πληροφοριακά συστήματα παρουσιάζουν τα εξής ιδιαίτερα χαρακτηριστικά (Fowler 2003):
  - Παραμένοντα δεδομένα (*persistent data*).
  - Μεγάλος όγκος δεδομένων που απαιτεί ειδικούς μηχανισμούς αποθήκευσης και συχνά καθορίζει την αρχιτεκτονική του συστήματος.
  - Ταυτόχρονη πρόσβαση στο σύστημα.
  - Αυξημένες απαιτήσεις επικοινωνίας με το χρήστη.
  - Επικοινωνία με άλλα πληροφοριακά συστήματα.
  - Ασφάλεια (*security*), έλεγχος (auditing), ταυτοποίηση (*authentication*), εξουσιοδότηση (*authorisation*)

Types of ISs (w.r.t Business Mgmt level)

Level of decision making	Focus of decision making	Typical IS applications	Typical IT solutions	Pivotal concept
Strategic (executive and senior management levels)	Strategies in support of organizational long-term objectives	Market and sales analysis, Product planning, Performance evaluation	Data mining, Knowledge management	***Knowledge***
Tactical (line management level)	Policies in support of short-term goals and resource allocation	Budget analysis, Salary forecasting, Inventory scheduling, Customer service	Data warehouse, Analytical processing, Spreadsheets	***Information***
Operational (operative management level)	Day-to-day staff activities and production support	Payroll, Invoicing, Purchasing, Accounting	Database, Transactional processing, Application generators	***Data***

 Chapter 1 (Maciaszek - RASD 2/e) 19

Information Systems Technologies (Τεχνολογικό υπόβαθρο των Πληρ. Συστημάτων)

- Database Management Systems (DBMS)
- Data Warehouses
- Data Mining
- Web technologies (HTML/XML, Web services)
- Information Retrieval Systems
- Communication technologies

Software Development: distinctive characteristics

- Η πολυπλοκότητα του πεδίου του προβλήματος
- Τα νοητικά χάσματα μεταξύ των εμπλεκομένων (πελατών, πωλητών, αναλυτών, προγραμματιστών, διοίκησης, κ.ά.).
- Το λογισμικό δεν είναι απτό
- Το λογισμικό δε φθείρεται
- Η ευελιξία που προσφέρεται από το λογισμικό.
- Η δυσκολία της διαχείρισης της διαδικασίας παραγωγής λογισμικού

- Software is a product of a creative act of development
  - a craft or an art in the sense of that activity performed by by an artisan rather than a fine artist
  - In a typical state of affairs, software is not a result of a repetitive act of manufacturing.

Software development invariants

- Complexity
  - Software is inherently complex
- Constraints
  - Software must conform to hardware/software platform, pre-existing ISs.
- Ability to change
  - Software must be build to accommodate change
- Invisibility
  - Software is buried deeply in “invisible” programming statements, binary library code, and surrounding system software.

Τεχνολογία Λογισμικού (Software Engineering)

- είναι μία επιστήμη του τεχνητού.
- δε συμπίπτει με την Επιστήμη των Υπολογιστών (*Computer Science*).

Οι Επιστήμες του Τεχνητού (*the Sciences of the Artificial*) [Simon1996]:

- Τα τεχνητά αντικείμενα
  - συνθέτονται (όχι πάντα σκόπιμα) από ανθρώπους
  - μπορεί να μιμούνται την εμφάνιση φυσικών αντικειμένων χωρίς όμως να μοιράζονται την ουσία τους
  - μπορούν να χαρακτηριστούν ανάλογα με τη λειτουργία τους, το σκοπό τους, και την προσαρμογή τους.
- Όταν σχεδιάζουμε τεχνητά αντικείμενα ασχολούμαστε με πώς πρέπει να είναι τα πράγματα, και όχι για το πώς είναι
- Ο επιστήμονας προσπαθεί να ερμηνεύσει μια κατάσταση, ο μηχανικός να κατασκευάσει.

Τεχνολογία Λογισμικού (Software Engineering)

- Η Τεχνολογία Λογισμικού:
  - ασχολείται με τεχνικές, μεθόδους και εργαλεία που βελτιώνουν την παραγωγή λογισμικού
  - ακολουθεί βήματα άλλων, ωριμότερων κλάδων, ώστε να βρεθούν και να υιοθετηθούν οι κατάλληλες, για το αντικείμενο του λογισμικού, τεχνικές και μεθοδολογίες.

Γιατί η τεχνολογία λογισμικού είναι σημαντική;

- Η οικονομία όλων των ανεπτυγμένων κρατών βασίζεται σε λογισμικό
  - Οι δαπάνες για ανάπτυξη του αποτελούν σημαντικό ποσοστό του ΑΕΠ αυτών των χωρών
- Ολοένα και περισσότερα συστήματα ελέγχονται από λογισμικό
- …. εξάρτηση από το λογισμικό

Ποια η διαφορά μεταξύ (α) Τεχνολογίας Λογισμικού και (β)Ανάλυσης και Σχεδίασης Πλ. Συστημάτων;

- Η (β) εστιάζει στα Πληροφοριακά Συστήματα
  - (όχι σε κάθε είδους λογισμικό)
- Κατά συνέπεια η θεματολογία της περιλαμβάνει:
  - Επιχειρηματικές Ανάγκες
  - Ανάλυση Σκοπιμότητας
  - Τεχνολογία Απαιτήσεων
    - Εκμαίευση, Συλλογή, Οργάνωση, Ανάλυση
  - Σχεδιασμός Διαχείρισης Δεδομένων (ER diagrams, Database Design)
  - Σχεδιασμός Αλληλεπίδρασης με Χρήστη
  - Μοντελοποίηση με UML
  - Χρήση εργαλείων CASE (Computer Aided Software Engineering)

So why do Analysis and Design ?

- Γιατί κάνουμε ένα σχεδιάγραμμα πριν γράψουμε μια έκθεση;
- Γιατί σχεδιάζουμε ένα σπίτι πριν το κτίσουμε;
- Γιατί σχεδιάζουμε έναν δρόμο πριν μπούμε στις μπουλντόζες;
- Γιατί σχεδιάζουμε ένα αυτοκίνητο πριν αρχίζουμε τη συναρμολόγηση;

- Ανάλυση και σχεδίαση ακόμα και στα … έργα τέχνης:
  - κινηματογράφος (storyboarding), λογοτεχνία (π.χ. Σταύρος Κρητιώτης, "Το μηνολόγιο ενός απόντος", Πόλις), θέατρο, γλυπτική, φωτογραφία, κλπ

Why do Analysis and Design ?
The current status in software engineering

The Spandish Group report, 2003:

- only one out of three software projects complete on-time and on-budget.
- 42% of all corporate IS projects were abandoned before completion

Why do Analysis and Design ?
The current status in software engineering

- Most errors (54%) are detected after coding and testing.
- Almost half of all errors (45%) are introduced during requirements and design.
- Most errors made during requirements analysis are non-clerical (77%)
- Requirements errors can cost up to 100 times more to fix than implementation errors - if they are not caught early on

Many failed systems were abandoned because analysts tried to build wonderful systems without understanding the organization.

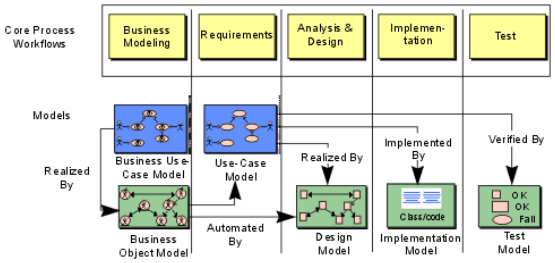
The primarily goal is to **create value for the organization**.

==> Need to do requirements and design right!

So why do Analysis and Design?

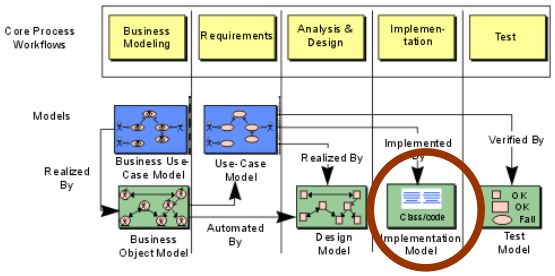
- Εφικτό;
- Εκτίμηση κόστους/χρόνου.
- Αποφυγή λαθών.
- Μείωση χρόνου/κόστους (ή αλλιώς, μεγιστοποίηση κέρδους)
- Εντοπισμός κινδύνων και πλάνο αντιμετώπισης τους
- Σειρά/στάδια κατασκευής

What is the result of Analysis and Design ?



- A bunch of
  - Notes
  - Diagrams that model various aspects of the systems
  - Tables
  - Figures
  - work plan, schedules
  - …
- UML is a standard way to organize all these

So why do Analysis and Design ?

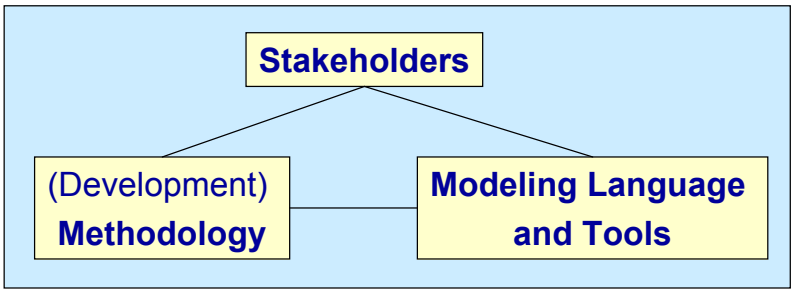


However:

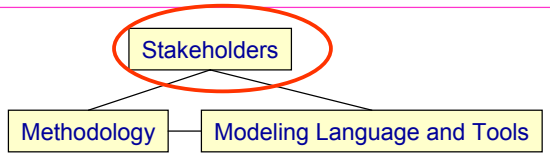
- The real point of software development is executable code
  - diagrams are, after all, just pretty pictures
  - no user is going to thank you for pretty pictures; what a user wants is software that executes
- So we must ask ourselves
  - why we are using UML?
  - How it will help us when it comes down to writing the code ?

Software Development: The triangle of success/failure

- Who is responsible for the success/failure of software engineering?



A. Stakeholders: The Players



- People that have a stake in a software project:
  - Customers (users and system owners)
  - Developers (analysts, designers, programmers, etc)
- Information systems are social systems:
  - developed by people (developers) for people (customers)
- The main causes of software failure can be traced to the stakeholder factor
  - on the customer end, and
  - on the developer end

A. Stakeholders: The Players

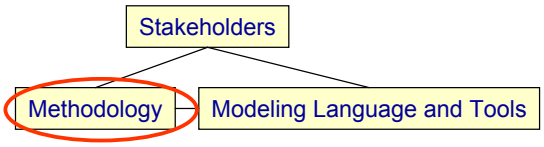
- Customers
  - owners
  - managers
  - users
- Developers
  - Analysts
  - Project manager
  - Designers
    - db designers
    - UI designers
  - Programmers

A. Stakeholders: The (roles of) Developers

Types of skills
- Technical
- Business
- analytical
- interpersonal
- management
- ethical

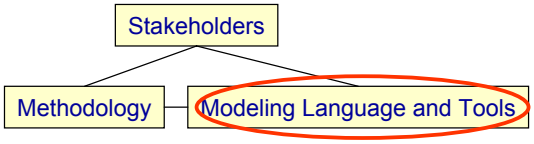
- Business analyst
  - analyses the key business aspects of the system
  - identifies how the system will provide business value
  - designs the new business processes and policies
- System analyst
  - Identifies how technology can improve business processes
  - designs the new business processes
  - designs the information system
  - ensures the project conforms to information systems standards
- Infrastructure analyst
  - ensures the system conforms to infrastructure standards
  - identifies infrastructure changes needed to support the system
- Change management analyst
  - develops and executes a change management task
  - develops and executes a user training plan
- Project manager
  - manages the team of analysts, programmers, technical writers, other persons
  - develops and monitors the project plan
  - assigns resources
  - serves as the primary point of contact for the project

B. Methodology
(or development methodology, software Process)



- Defines activities and organizational procedures used in software production and maintenance
- A process model (methodology):
  - states an order of carrying out activities
  - specifies what development artefacts are to be delivered when
  - assigns activities and artefacts to developers
  - offers criteria for monitoring a project's progress, for measuring the outcomes, and for planning future projects
- Is not susceptible to standardisation

C. Modelling Language and Tools



Modelling artefacts have to be communicated and documented.

- UML (Unified Modeling Language)
  - general purpose visual modelling language that is used to specify, visualise, construct, and document the artefacts of a software system
- CASE (Computer-Assisted Software Engineering) tools
  - enables storage and retrieval of models in a central repository and graphical and textual manipulation of models on a computer screen

Information Systems Methodologies

- Where do we start ? ===> Feasibility study
- Define the problem ===> Requirements analysis
- Define a solution =====> Design

This course is about methodologies for building information systems