LINGI 2255

2017–2018
Prof. Kim Mens

moodleucl.uclouvain.be
Course: LINGI2255

LINGI 2255

Software Life Cycle

Software Development Process Phases

Requirements

Implementation

Design

Initial Planning

Deployment

Architecture

Testing

Evaluation

Maintenance

The Waterfall Model

The "waterfall" model sees the software life-cycle as a step-by-step waterfall between the various development phases.

Some disadvantages of the waterfall model:

- requirements "frozen" early in the life-cycle
- requirements are validated too late

Iterative and Incremental Development

- A more cyclic software development process
- Overcomes some weaknesses of waterfall model
- Essential part of
  - RUP (Rational Unified Process)
  - XP (extreme programming)
  - agile software development

Iterative and Incremental Development

LINGI 2255

Software Requirements

Software Requirements

- Requirements phase :
  - the phase in the software life-cycle where it is defined **what the system should do**, i.e., what are the (functional and non-functional) requirements?
- Requirement :
  - a statement about what the proposed system will do that all stakeholders agree must be made true in order for the **customer's problem** to be adequately solved [Lehtbridge&Laganiere2001]

The Importance of Software Requirements

Rates of Software Engineering Failure

Requirements	Very High
Specification	Low
Design	Low
Implementation	Low
Installation	High
Operation	Enormous
Maintenance	Very High

The Importance of Software Requirements

- Getting the requirements wrong is one of the main reasons of failure of software projects
- **Building the right system** ≠ building the system right
- Identifying the requirements is important but hard
  - Questioning the **customer** helps in this phase
  - **Artefacts** can be built to communicate requirements

The Importance of Software Requirements

- Project team delivers a product without having a clear understanding of what the customer wants
- The customer does not like it because the requirements have not been met
- This leads to project rejection or failure

http://spectrum.ieee.org/computing/software/why-software-fails

LINGI 2255

Activity Diagram

Activity Diagram

- models **computational** or organisational **processes**
  - high-level view of behaviour and flow of software system
- show the **overall flow** of control
  - kind of flowchart; with choice, iteration and concurrency
- graphical representation of **stepwise activities** and actions

Activity Diagrams

- can be used during requirements analysis
  to provide a high-level view of the system behaviour
  - non-technical : can be understood by end-users
  - focus on high-level activities : no implementation details
  - shows how control flows from one activity to another
- note: you can also use an activity diagram to describe the organisational process of how your team will work

An Activity Diagram

- Example of computational activity diagram taken from http://www.agilemodeling.com/artifacts/activityDiagram.htm

An Activity Diagram

An Activity Diagram

Final node (*filled circle with border*): ending point of the diagram

Initial node (*filled circle*) : starting point of diagram

An Activity Diagram

An **activity** representing an electronic activity (showing a screen)

An **activity** (*rounded rectangle*), here representing a physical activity (inspecting forms)

An Activity Diagram

A **decision** (*diamond* with one flow entering and several leaving)

A **condition** which must be true in order to traverse this arrow

In a **merge**, flows come together before continuing

A **flow / edge** (*arrow*) describes flow of the diagram

An Activity Diagram

A **join** (*black bar* with several flows entering and one leaving) denotes the end of parallel processing.

A **fork** (*black bar* with one flow entering and several leaving) denotes the start of a parallel activity

Activity Diagrams

Sub-activity indicator means that the activity is described in a more detailed activity diagram

Partitions or swimlanes indicating who/what is performing the activities (either the Applicant, Registrar, or System)

Activity Diagrams

Flow final (*circle with X through it*) indicates that the process stops at this point

An Activity Diagram

- Example of organisational activity diagram taken from https://en.wikipedia.org/wiki/Activity_diagram

Conducting
[else]
[inexperienced participants in the group]
Explain problem
Warm-up practice
Present rules
Call for ideas
[idea(s) available]
[more]
Participants write down their idea
[no idea(s)]
[one]
Suggest lead
Everybody presents their idea in turn
Present idea
Record idea
[no time left]
[time left]
Elaborate on idea
[one]
[more]
[none]
[time left]
Select most associated idea
Others write down their idea
[no time left]
Wrap up

Activity Diagrams

- For more information on activity diagrams, consult :
  - http://en.wikipedia.org/wiki/Activity_diagram
  - http://www.agilemodeling.com/artifacts/activityDiagram.htm
  - http://www.tutorialspoint.com/uml/uml_activity_diagram.htm
  - http://sourcemaking.com/uml/modeling-business-systems/external-view/activity-diagrams
  - http://www.sparxsystems.com.au/resources/uml2_tutorial/uml2_activitydiagram.html

LINGI 2255

Use Case

Use Case

- describes a way in which a real-world actor interacts with the software system
  - captures scenarios the system should be able to handle
- written in brief, bullet points in natural language
  - list of actions or event steps
- contains just enough information to get the idea across
  - advice: keep your models as simple as possible and only document in more detail if this adds actual value

Example

from: http://agilemodeling.com/artifacts/systemUseCase.htm

Name: Enroll in Seminar

Identifier: UC 17

Basic Course of Action:

- Student inputs her name and student number
- System verifies the student is eligible to enroll in seminars. If not eligible then the student is informed and use case ends.
- System displays list of available seminars.
- Student chooses a seminar or decides not to enroll at all.
- System validates the student is eligible to enroll in the chosen seminar. If not eligible, the student is asked to choose another.
- System validates the seminar fits into the student's schedule.
- System calculates and displays fees
- Student verifies the cost and either indicates she wants to enroll or not.
- System enrolls the student in the seminar and bills them for it.
- The system prints enrollment receipt.

LINGI 2255

Usage Scenario

A Usage Scenario

- describes a real-world example of how one or more persons or organisations interact with a system
- describe the steps, events, and/or actions which occur during the interaction
  - describing at a high-level *what* happens without indicating *how* these steps are performed

Example

- Example taken from http://www.agilemodeling.com/artifacts/usageScenario.htm

Scenario: ATM banking for the week.

1. Sally Jones places her bank card into the ATM.
2. Sally successfully logs into the ATM using her personal identification number.
3. Sally deposits her weekly paycheck of $350 into her savings account.
4. Sally pays her phone bill of $75, her electric bill of $145, her cable bill of $55, and her water bill of $85 from her savings account
5. Sally attempts to withdraw $100 from her savings account for the weekend but discovers that she has insufficient funds
6. Sally withdraws $40 and gets her card back

A Usage Scenario

- as opposed to use cases, usage scenarios refer to examples or real actors, as opposed to generic actors
- personalising the scenarios increases their understandability
- also, usage scenarios typically describe a single path of logic
- whereas use cases often describe several paths (the basic path plus any appropriate alternative paths)

LINGI 2255

User Story

User Story

- a way of capturing requirements
  - often used with “agile” development approaches
- a short, high-level, description of small bit of functionality
  - that fits on an index card

User Story Template

- a simple, descriptive title
- a short description, expressed as a phrase of the form:
  - As a < ***role*** >,
    I want to < ***goal*** / *desire* >
    so that I < ***benefit*** >
  - In order to < *receive* ***benefit*** >
    as a < ***role*** >,
    I want to < ***goal*** / *desire* >

Example

Edit Homepage

*As a **site administrator**,*
*I want to be able to **edit the homepage***
*so that I can **update the user testimonials***

Advantages of this User Story Template

- The "as a" phrase indicates what kind of user is involved in the story.
- With the "I want" phrase, a person identifies more closely with the story.
- The "so that" phrase helps to estimate the story's importance and relevance, by focussing on *why* we want the goal to be achieved.

What is a Good User Story ?

- INVEST
  - "Independent"
  - "Negotiable"
  - "Valuable" (of value for the user and customer)
  - "Estimable"
  - "Small"
  - "Testable"

User Stories

- Can describe not only functional but also **non-functional** requirements (security, performance, ...):
  - As a customer, I want to be able to run the software on all versions of Windows from Windows 7 onwards.
  - As the CTO, I want the system to use our existing orders database rather than create a new one, so that we don't have yet another database to maintain.
  - As a user, I want the site to be available 99% of the time I try to access it, so that I don't get frustrated and find another site to use.
  - As someone who speaks a Latin-based language, I might want to run your software someday.

LINGI 2255

User Story

vs.

Use Case / Usage Scenario

User Story vs. Use Case / Usage Scenario

- User stories and use cases / usage scenarios serve more or less the same purpose
  - to capture specific user requirements in terms of interactions between the user and the system
- But they are not the same

User Story vs. Use Case / Usage Scenario

- User stories are typically much smaller than other usage requirement artefacts such as use cases or usage scenarios
- User stories are more centred on the goals and benefits of the functionality you're describing
- Should help the reader understand what the software should accomplish
- Whereas use cases are more granular, and describe in more detail how your system will act

LINGI 2255

Use Case Diagram

Use Case Diagram

- simple representation of how and which users interact with a software system
- in conjunction with more textual descriptions of individual use cases
- provides higher-level view of the system as a whole by providing a simplified graphical representation of what the system is supposed to do
- each particular use case drills into more detail about every possibility

Example

Input student marks
Grade Admistrator
Obtain Student Grant
Obtain Student Loan
Reimburse Course Fees
Pay Fees
Financial Institution
Print Teaching Schedule
Teach Seminar
Instructor
Student
Distribute Transcripts
<<extend>>
Distribute Information to Students
Post Office
Distribute Fee Schedule
<<extend>>
<<extend>>
Distribute Schedules
Enroll in seminar
Drop seminar
Registrar
Drop out of School
Attend seminar
Graduate From School
Finish seminar
Apply for Grant
Researcher

LINGI 2255

WIREFRAME

Wireframes

- commonly used to prototype content, lay-out and functionality of pages in a web application
- visual illustrations of a web page
- meant to show all items included on a page without defining the look and feel or graphic design yet
- meant to illustrate the features, content and links that need to appear on a page

Wireframing

- typically done early in a project
  - after requirements have been elicited, so that the functionality is known
  - early enough so that feedback from customer or end-user can be gained before the actual design and implementation is started
- Cf. http://en.wikipedia.org/wiki/Website_wireframe

Example drawn in Excel

Home | Concept | FAQ | News | Contact

about
etc.

disclaimer etc.

Example drawn by hand

Example drawn by tool

LINGI 2255

Requirements Document

Deliverable D4 : Requirements Document

- **Friday October 6, 2017 at 6pm (intermediate)**
  - Usage scenarios and user stories
- **Friday, October 13, 2017 at 6pm (deliverable)**
  - Full document, no more than 10 pages
  - counts for 10% of final score
- Focus on the essence (no “reporting because of the reporting”)

Deliverable D4 : Requirements Document

- **detailed description** of selected module to be implemented by your team
- a representative set of **user stories** describing the functional and non-functional requirements of that module
- one or two **usage scenarios** illustrating key functionality of your module
- some **mock-ups** or **wireframes** of the module to be implemented

Deliverable D4 : Requirements Document

- **activity diagram**(s) describing high-level view of behaviour and flow of the software system
- **initial planning** with indication of **deliverables**
- **development methodology** to be adopted (waterfall, agile, hybrid, other)
- **team** number, members, leader, assistant
- organisation, distribution of tasks, roles

Remember...

Requirements document forms **contract** with customer

- should describe what customer expects from software system to be delivered
- should be understandable by the customer
- you will need to account later on for requirements not achieved
- if some requirements are optional, indicate this carefully

Attribution-ShareAlike 4.0 International (CC BY-SA 4.0)

You are free to:

Share — copy and redistribute the material in any medium or format

Adapt — remix, transform, and build upon the material for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

https://creativecommons.org/licenses/by-sa/4.0/