SOFTWARE PROJECT

LECTURE 1

WOUTER SWIERSTRA

TRY TO SIT WITH THE REST OF YOUR

SO YOU THINK YOU CAN PROGRAM...

INTENDED LEARNING OUTCOMES

AT THE END OF THIS COURSE, YOU SHOULD BE ABLE TO

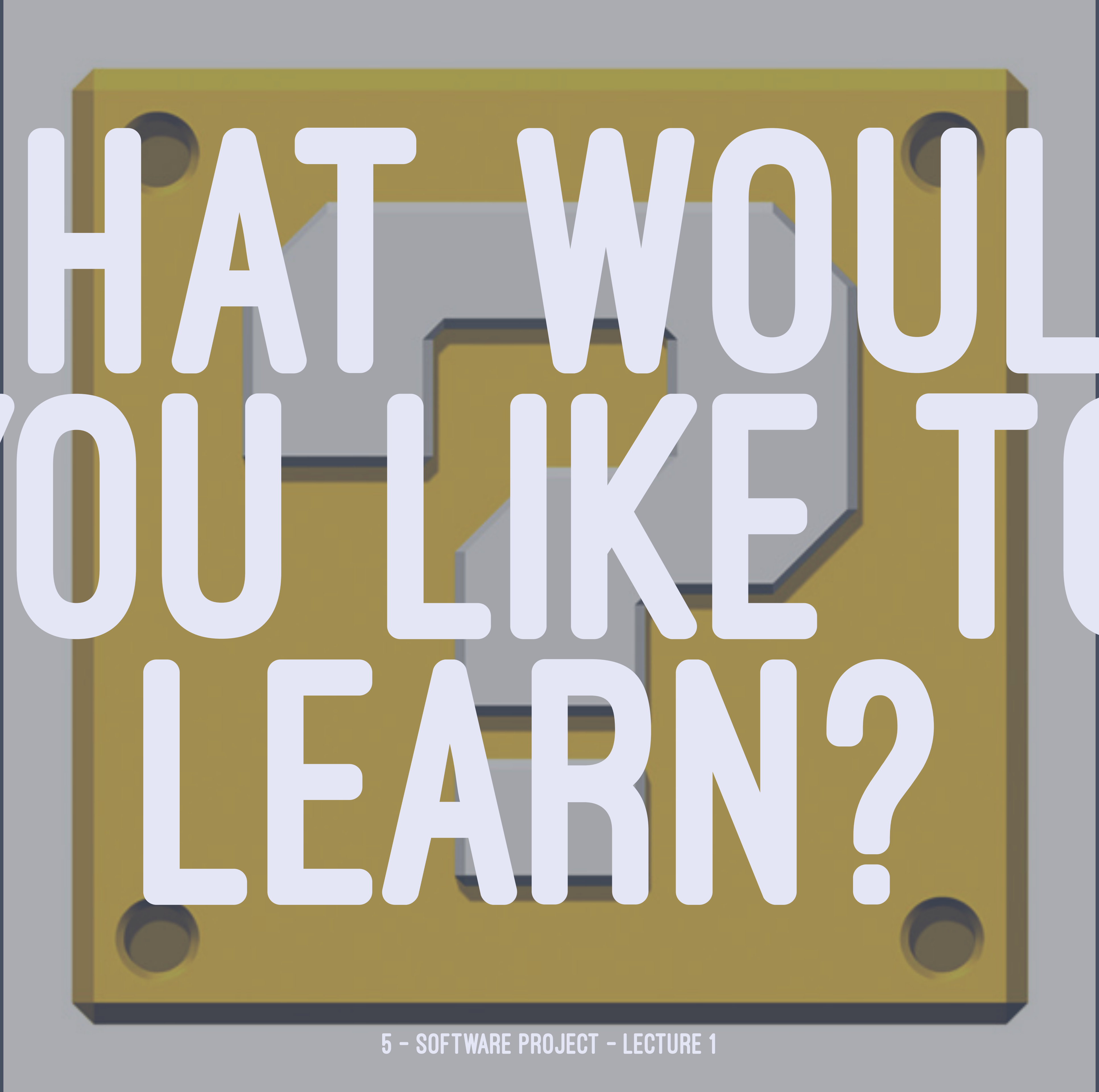
- WORK IN TEAM
- INTERACT WITH CUSTOMERS
- TRANSLATE CUSTOMER REQUIREMENTS TO A WORKING PRODUCT
- PLAN AND DEVELOP A SOFTWARE APPLICATION

THEORY & PRACTICE

- YOU HAVE LEARNED ABOUT THE THEORY OF SOFTWARE DEVELOPMENT IN VARIOUS COURSES YOU TAKE:
- THE SOFTWARE PROJECT GIVES YOU A VALUABLE OPPORTUNITY TO PUT THESE IDEAS INTO PRACTICE.

THESE LECTURES TRY TO BRIDGE THE GAP BY PROVIDING PRACTICAL ADVICE FOR YOUR PROJECT.

WHAT WOULD YOU LIKE TO LEARN?



TOPICS COVERED

DIFFERENT PARTS OF SOFTWARE ENGINEERING:

- SOFTWARE PROJECT MANAGEMENT
- RISK & ARCHITECTURE
- COLLABORATIVE SOFTWARE DEVELOPMENT

SOFTWARE PROJECT MANAGEMENT (SCRUM & AGILE)

- HOW TO ORGANIZE A TEAM OF DEVELOPERS?
- HOW TO PLAN DEVELOPMENT?
- PRESENT YOUR TEAM STRUCTURE, PRODUCT BACKLOG & RISKS.

SOFTWARE REQUIREMENTS & ARCHITECTURE

- HOW TO IDENTIFY RISK?
- WHAT IS SOFTWARE ARCHITECTURE?
- HOW DO RISKS AND QUALITY ATTRIBUTES INFLUENCE MY ARCHITECTURE?
- PRESENT THE KEY RISKS AND SOFTWARE ARCHITECTURE OF YOUR PROJECT.

SOFTWARE DEVELOPMENT

- HOW CAN TO USE VERSION CONTROL EFFECTIVELY?
- HOW TO DEPLOY SOFTWARE AUTOMATICALLY? OR SCRIPT THE BUILD PROCESS?
- HOW TO TEST COMPLEX SOFTWARE?
- WHAT IS TEST-DRIVEN DEVELOPMENT? WHY IS IT IMPORTANT?
- PRESENT YOUR QA METHODOLOGY.

THE CHALLENGE IN TEACHING THIS COURSE

HOW TO TEACH PRINCIPLES WITHOUT BEING VAGUE?

HOW TO TEACH SKILLS WITHOUT BEING PROJECT-SPECIFIC?

SOFTWARE PROJECT MANAGEMENT

PLAN DEVELOPMENT IN YOUR HOBBY PROJECTS.

THE MARSHMALLOW CHALLENGE

RULES

MAKE A FREESTANDING STRUCTURE OUT OF SPAGHETTI, TAPE AND STRING WITH THE MARSHMALLOW ON TOP.

- IT CANNOT BE ATTACHED TO THE CEILING OR OTHER HIGHER OBJECTS. NO HOLDING THE STRUCTURE.
- I'LL MEASURE THE DISTANCE FROM THE TABLE TO THE BOTTOM OF THE MARSHMALLOW.
- FEEL FREE TO BREAK THE SPAGHETTI, TAPE, OR ROPE. THE

RULES

- YOU HAVE 18 MINUTES.
- TEAM WITH THE HIGHEST STRUCTURE WINS THE REST OF THE BAG OF MARSHMALLOWS.
- ANY QUESTIONS?

REFLECTION

THE MARSHMALLOW CHALLENGE

REFLECTION

WHAT I HOPE TO TEACH

- THE IMPORTANCE OF IDENTIFYING RISK
- ITERATIVELY REFINE A WORKING PROTOTYPE
- NONE OF YOU ARE ARCHITECTS.

SOFTWARE PROJECT MANAGEMENT

THE SOFTWARE DEVELOPMENT PROCESS

THE SOFTWARE DEVELOPMENT PROCESS IS THE SET OF ACTIVITIES THAT PRODUCE A PIECE OF SOFTWARE:

- REQUIREMENTS ENGINEERING & WRITING SPECIFICATIONS;
- ANALYSIS AND DESIGN;
- SOFTWARE VALIDATION AND TESTING;
- MAINTAINANCE AND EVOLUTION.

SOFTWARE DEVELOPMENT METHODS

- A SOFTWARE DEVELOPMENT METHOD IS AN ORGANIZED WAY OF CONSTRUCTING SOFTWARE - AN 'ALGORITHM' FOR SOFTWARE DEVELOPMENT
  - IDEA: DOCUMENTING AN APPROACH WILL MAKE SUCCESS REPEATABLE, PROJECTS (COST-)EFFECTIVE, AND RESULTS PREDICTABLE.

KNOWING VERSUS DOING

MANY OF YOU MAY KNOW THE THEORY BEHIND STRUCTURED SOFTWARE DEVELOPMENT PROCESSES.

BUT ALMOST NONE HAVE APPLIED THESE IDEAS WELL.

DELIVERING THE RIGHT PIECE OF WORKING SOFTWARE ON TIME IS HARD.

CHOICES, CHOICES, CHOICES!

WIKIPEDIA HAS A LIST OF DIFFERENT SOFTWARE DEVELOPMENT METHODS

WHICH IS BEST FOR YOU?

CHARACTERISTICS OF THESE SOFTWARE/ GAMETECH

THE SOFTWARE PROJECT

- LIMITED RESOURCES: TIME, PEOPLE, MEANS.
- EQUALLY EXPERIENCED DEVELOPERS
- COMMITTED CUSTOMERS: NOT MANY TROUBLESOME STAKEHOLDERS
- TYPICALLY, NO HUGE EXISTING CODEBASE.

SCRUM

THE AGILE MANIFESTO (2001)

WE ARE UNCOVERING BETTER WAYS OF DEVELOPING SOFTWARE BY DOING IT AND HELPING OTHERS DO IT. THROUGH THIS WORK WE HAVE COME TO VALUE:

- INDIVIDUALS AND INTERACTIONS OVER PROCESSES AND TOOLS
- WORKING SOFTWARE OVER COMPREHENSIVE DOCUMENTATION
- CUSTOMER COLLABORATION OVER CONTRACT NEGOTIATION

WHAT IS AGILE?

AGILE SOFTWARE DEVELOPMENT REFERS TO A COLLECTION OF SOFTWARE DEVELOPMENT METHODS:

- SCRUM
- XTREME PROGRAMMING
- KANBAN
- AND MANY OTHERS

THEY ARE ALL THE AGILE METHODS

THE SCRUM FRAMEWORK

- ONE POPULAR APPROACH ORGANISING AN AGILE DEVELOPMENT TEAM.
    - DEVELOPED BY JEFF SUTHERLAND IN 1993.
- GIVEN THE TIME FRAME AND TYPE OF PROJECT, SCRUM MIGHT BE A GOOD OPTION FOR YOU.

THE SCRUM TEAM

THERE ARE THREE ROLES IN THE SCRUM:

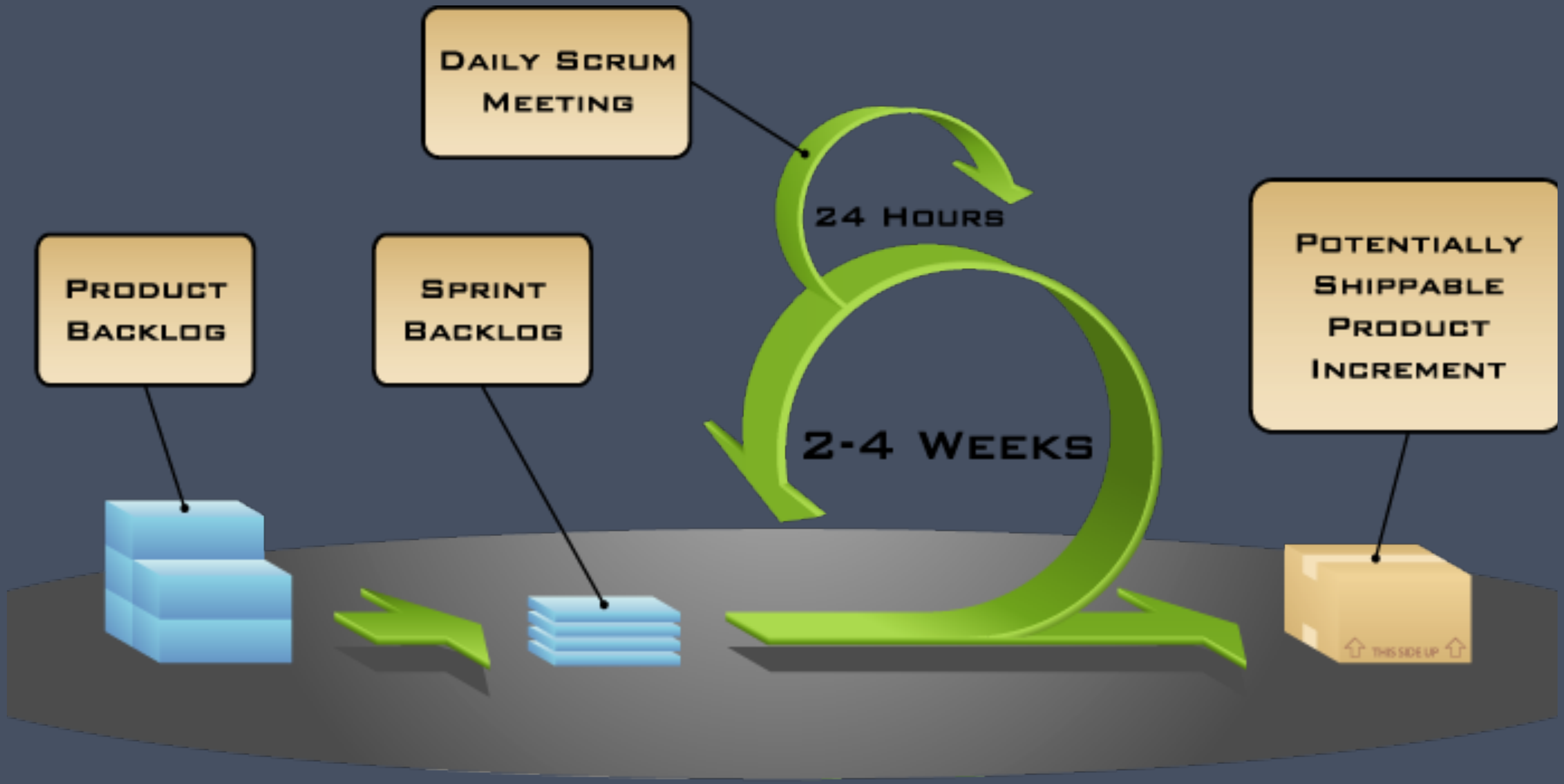
- THE PRODUCT OWNER
- THE SCRUM MASTER
- THE DEVELOPMENT TEAM
- (YOUR UU SUPERVISOR)

THE SCRUM PROCESS - I

- TOGETHER WITH THE PRODUCT OWNER, YOU WILL CREATE A 'WISHLIST OF FEATURES', CALLED THE PRODUCT BACKLOG
- THE DEVELOPMENT TEAM TAKES A SMALL NUMBER OF ITEMS FROM THE TOP OF THE BACKLOG: SPRINT BACKLOG.
- THESE ITEMS ARE COMPLETED IN A SINGLE ITERATION (2 WEEKS)

THE SCRUM PROCESS - II

- DAILY MEETINGS TO ASSESS PROGRESS
- AT THE END OF THE SPRINT, THE STORIES SHOULD BE COMPLETELY IMPLEMENTED. THERE SHOULD BE A SHIPPABLE PRODUCT INCREMENT READY.
- EVERY ITERATION A SPRINT REVIEW: HOW TO IMPROVE THE NEXT SPRINT.
- PLAN THE NEXT SPRINT AND REPEAT.


Daily Scrum Meeting
24 Hours
Product Backlog
Sprint Backlog
2-4 Weeks
Potentially Shippable Product Increment
This side up

THE PRODUCT OWNER

THE PRODUCT OWNER IS A SINGLE PERSON REPRESENTING ALL STAKEHOLDERS:

- HELPS PRIORITIZE THE PRODUCT BACKLOG
- RESPONSIBLE FOR PRODUCT VISION
- PRIORITIZES THE PRODUCT BACKLOG
- ACCEPTS OR REJECTS PRODUCT INCREMENTS

THE SCRUM MASTER

THE SCRUM MASTER IS RESPONSIBLE FOR MAKING SURE THE PROCESS IS SMOOTH:

- REMOVE OUTSIDE DISTRACTIONS;
- CHAIR REVIEW MEETINGS OR THE DAILY STANDUP;
- FACILITATES THE PROCESS, WITHOUT BEING A A PROJECT MANAGER.

WE WILL DISTINGUISH BETWEEN *SCRUM MASTER* AND *COACH*

THE DEVELOPMENT TEAM

- THAT IS YOU.
- NO SPECIFIC ROLES FOR TESTING, DESIGN, ETC.
- YOU ALL SHARE OWNERSHIP OF THE CODE.
- YOU ORGANIZE YOURSELF - ALL OUTSIDE INFLUENCE GOES THROUGH THE SCRUM MASTER AND PRODUCT OWNER.

SCRUM MASTER & COACH

RATHER THAN ASSIGN ALL 'MANAGEMENT' RESPONSIBILITY TO A SINGLE PERSON (THE SCRUM MASTER), WE DISTINGUISH TWO ROLES:

- SCRUM MASTER - MANAGES THE PRODUCT BACKLOG; HAS AN OVERVIEW OF SCHEDULING AND TEAM'S PERFORMANCE.
- COACH - CHAIRS STANDUPS, REVIEW AND PLANNING MEETINGS; ENSURES EVERYONE IS HEARD; ENSURES DECISION MAKING IS

STANDUP MEETINGS - I

- START EACH DAY WITH A STAND-UP MEETING.
- THE STAND UP MEETING IS AT A FIXED TIME AND LOCATION, REGARDLESS OF IF PEOPLE ARE MISSING.
- PRO-TIP: SET A REPEATING ALARM ON YOUR CELLPHONE.
- PRO-TIP: LATE? 1 EURO FINE.
- EVERYONE STANDS UP - NO SITTING!

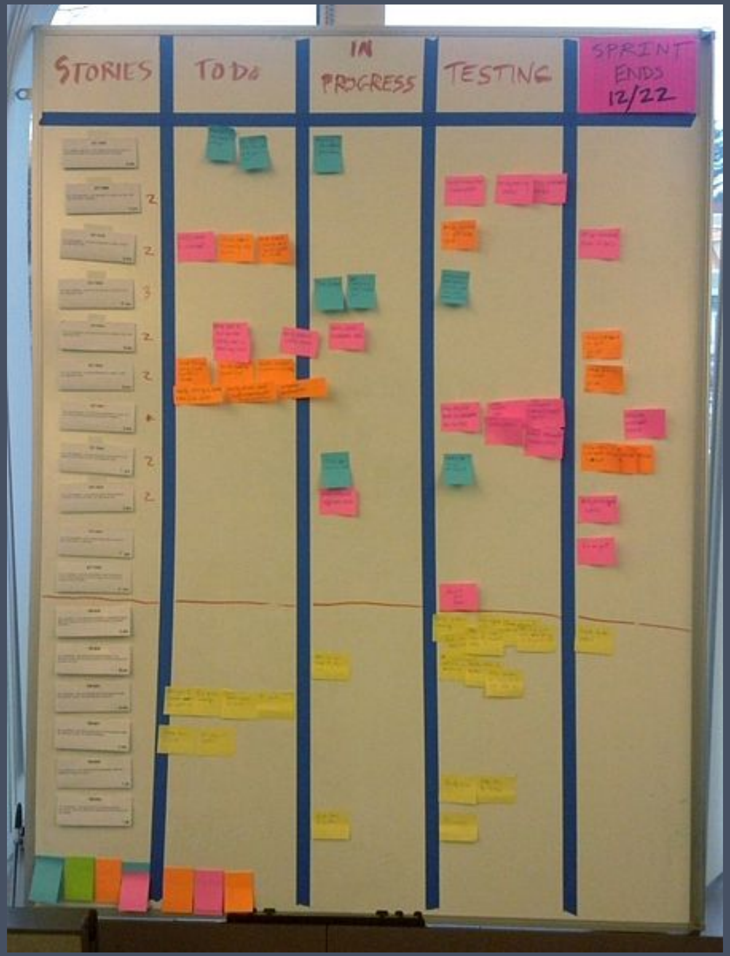
STANDUP MEETINGS - II

EVERYONE ON THE TEAM SAYS:

- WHAT HAVE I DONE YESTERDAY?
- WHAT PROBLEMS AM I FACING?
- WHAT AM I GOING TO DO TODAY?

DAILY MEETINGS HELP COORDINATE DEVELOPER ACTIVITY.

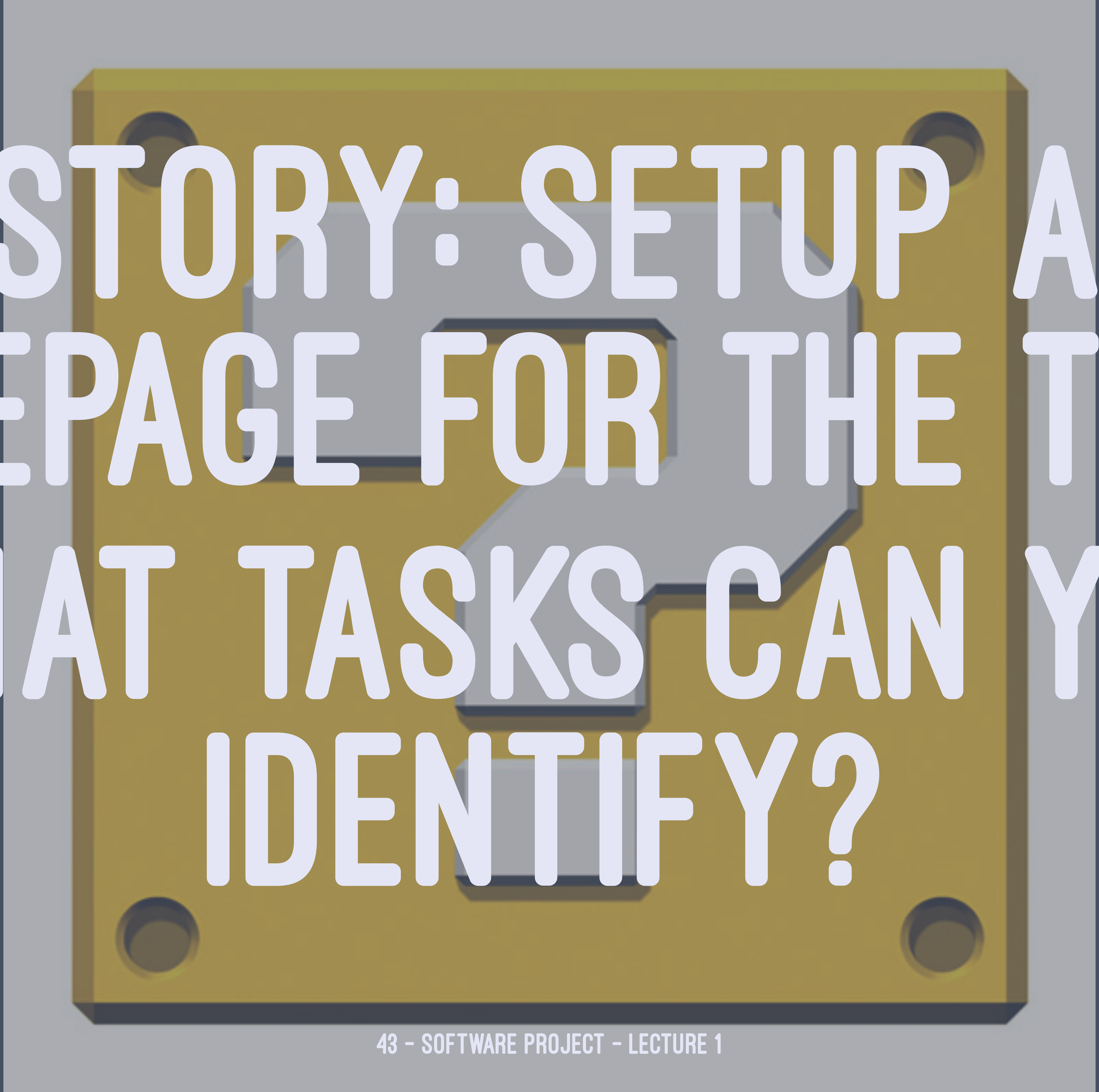
THE TASK BOARD



OR DIGITALLY USING TOOLS LIKE TRELLO

THE TASK BOARD - I

- WHEN PLANNING THE SPRINT, EVERY DEVELOPER IS ASSIGNED STORIES.
- EACH DEVELOPER PLANS THE IMPLEMENTATION OF THIS STORY, AND SPLITS THIS INTO SMALL TASKS.
- EACH TASK GOES ON A POST-IT NOTE.



STORY: SETUP A HOMEPAGE FOR THE TEAM. WHAT TASKS CAN YOU IDENTIFY?

EXAMPLE TASKS

- CREATE GITHUB REPOSITORY & ADD COLLABORATORS.
- PUT CONTENT ON BARE HTML PAGE.
- ADD STYLING WITH A CSS FILE.
- TEST ACROSS DIFFERENT BROWSERS, OPERATING SYSTEMS, AND DEVICES.
- PUBLISH LINKS ON SOCIAL MEDIA.

THE TASK BOARD - II

- THE TASKS INCLUDE TESTING, DEPLOYMENT, ETC.
- ALL THE POST-IT NOTES OF ALL STORIES GO ON A SINGLE BOARD, TOGETHER
WITH THE ORIGINAL STORY DESCRIPTION.

THE TASK BOARD - III

- ALL THE STORIES AND ASSOCIATED POST-ITS ARE HAVE THEIR OWN ROW.
- THE ROWS OF THE BOARD ARE DIVIDED INTO SEVERAL COLUMNS: TODO, IN PROGRESS, AND DONE. (OTHER VARIATIONS EXIST)
- UPDATE THE BOARD DURING EVERY STANDUP MEETING.
- AT THE END OF THE ITERATION, EVERYTHING IS DONE.

END OF ITERATION

- GIVE A DEMO TO THE STAKEHOLDERS AT THE END OF THE ITERATION.
- CLEAN THE TASK BOARD.
- CREATE NEW STORIES FOR ANY UNFINISHED WORK.

REGULAR REVIEW

- END EACH ITERATION WITH A RETROSPECTIVE:
  - WHAT WENT WELL?
  - WHAT CAN WE DO BETTER?

START PLANNING THE NEXT ITERATION AFTER THE DEMO. YOUR DEMO SHOULD HELP YOUR CUSTOMERS AND PRODUCT OWNER DECIDE WHAT THEY WANT TO SEE NEXT.

AGILE/SCRUM PITFALLS

- DON'T ADD STORIES DURING THE SPRINT.
- DON'T SOLVE PROBLEMS DURING THE STANDUP.
- LOTS OF DEADLINES MEANS LOTS OF CHANCES TO WRITE BAD CODE TO MEET A DEADLINE, INTRODUCING *TECHNICAL DEBT*.
- PRIORITIZING STORIES IS HARD.
- YOU ARE NOT EXPERIENCED DEVELOPERS.

TEACHING SOFTWARE DEVELOPMENT METHODS

LISTENING TO ME TALK ABOUT SCRUM IS EASY.

APPLYING THE AGILE PHILOSOPHY IS NOT.

IN MY EXPERIENCE, EVEN SMART PEOPLE STRUGGLE TO GET THIS RIGHT.

AGILE METHODS ARE STILL WELL-DEFINED METHODS.

BEING AGILE IS NOT AN EXCUSE FOR COWBOY CODING OR LACK OF

BEYOND SCRUM

THERE ARE A LOT OF OTHER AGILE METHODS:

- AGILE UNIFIED PROCESS:
- KANBAN
- LEAN SOFTWARE DEVELOPMENT
- SCRUM-BAN

YOU MAY HAVE EXPERIENCE WORKING WITH METHODOLOGY - USE

LEARN MORE

LOTS OF GREAT FREE MATERIAL ONLINE:

- THE OFFICIAL SCRUM GUIDE
- FREE SCRUM TRAINING VIDEOS: HTTP://SCRUMMETHODOLOGY.COM/
- THE SCRUM REFERENCE CARD
- THE SCRUM PRIMER

PLANNING & METHODOLOGY MAY NOT BE THE BIGGEST TECHNICAL CHALLENGE IN YOUR PROJECT: BUT IT HAS A **HUGE IMPACT** ON

SCRUM COACHING

FRIDAY FEBRUARY 13TH, MAX VERHORST (IT CONSULTANT AND SCRUM COACH AT INFOSUPPORT) WILL VISIT THE UNIVERSITY.

YOU WILL HAVE AN HOUR LONG SESSION WITH HIM TO PRESENT YOUR PROJECT,
PRODUCT BACKLOG, AND PLANS.

PLEASE SIGN UP FOR A SLOT!