System Development and Software Process

System Development Fundamentals

- Software systems should be developed using a managed and well-understood development process; different processes may be used for different types of software
- Some fundamental principles apply to all types of software systems, irrespective of the development techniques used
- Reliability and performance are important for all types of system

System Development Fundamentals

- Understanding and managing the software specification and requirements (what the software should do) is very important
- If appropriate, we should reuse software that has already been developed rather than write new one

System: a part of a larger activity

- Software systems are frequently developed to support a part (or parts) of the overall mission of an "enterprise"
- The term *enterprise* does not have to mean a business (it may be work on a new aircraft design, operations of a laboratory, or any other type of activity)
- Also true of mobile applications

From: www.oilandgasproductnews.com

Parties in System Development

Parties (stakeholders) involved in system development:

- Users (direct/indirect)
- Customer/procurer
- Producer/developer

The three parties usually include different groups of people!

Mobile development may be a bit different, but not by much

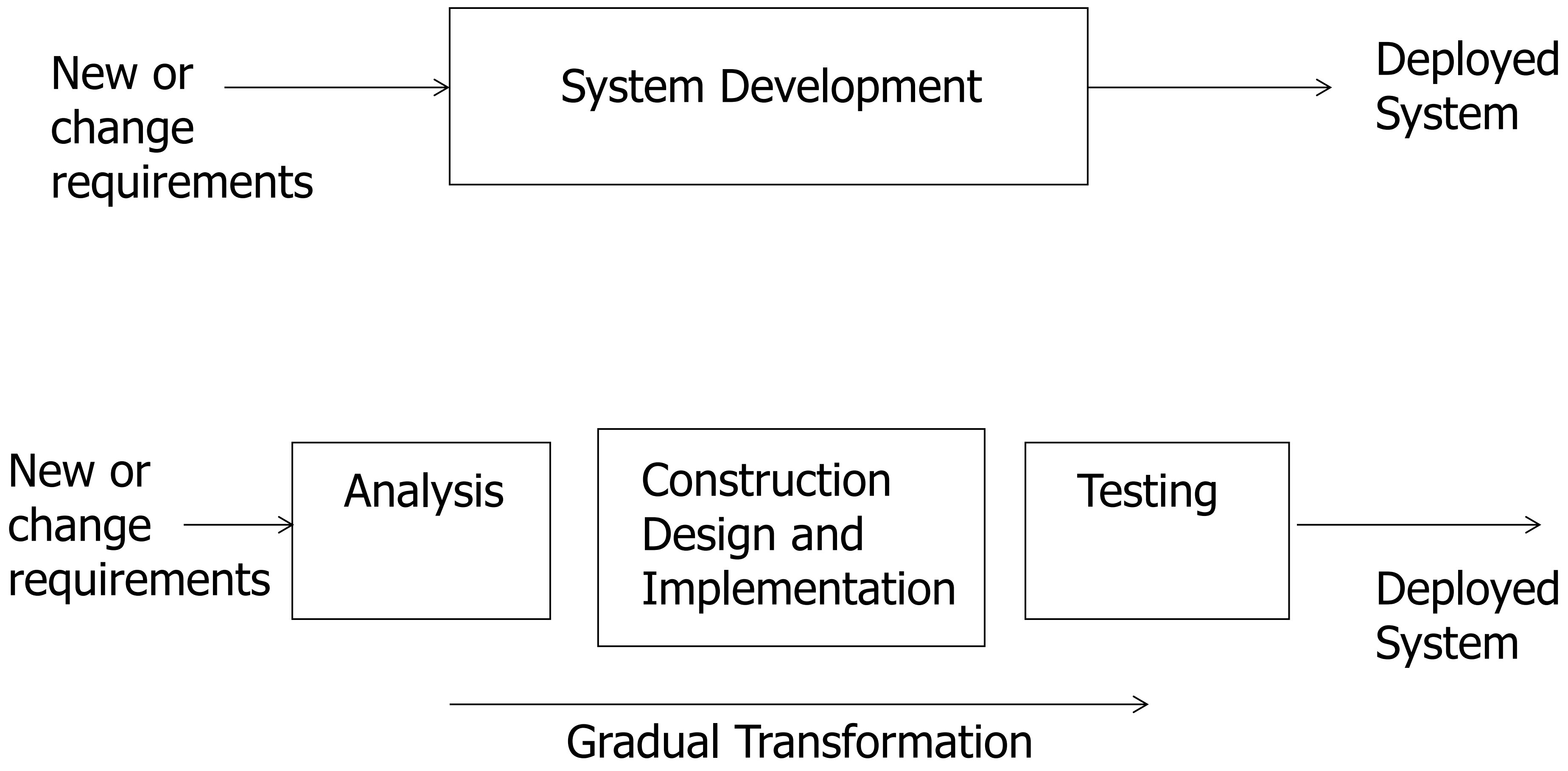
Parties in System Development

Users play a central role

A software system should be:

- specified based on user needs,
- validated whether it really functions according to user needs, and
- documented, by describing the system from the user’s perspective

Gradual Transformation in System Development


New or change requirements
System Development
Deployed System
New or change requirements
Analysis
Construction Design and Implementation
Testing
Deployed System
Gradual Transformation

Developing large systems

What is a large software system?

Developing large systems

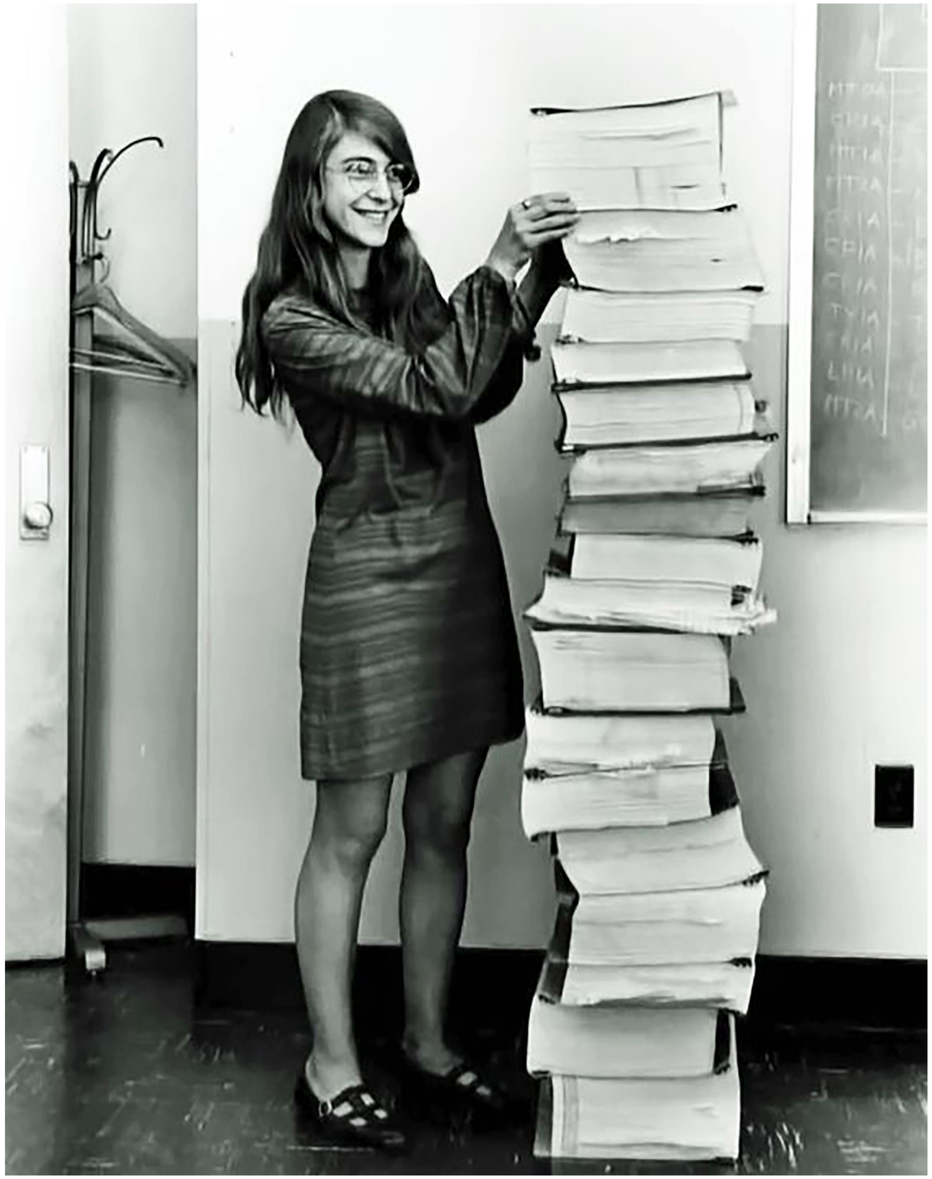
What is a large software system?

- Imagine a system where you print your code and it’s ***11k pages*** (this is well over half a million lines of code)!

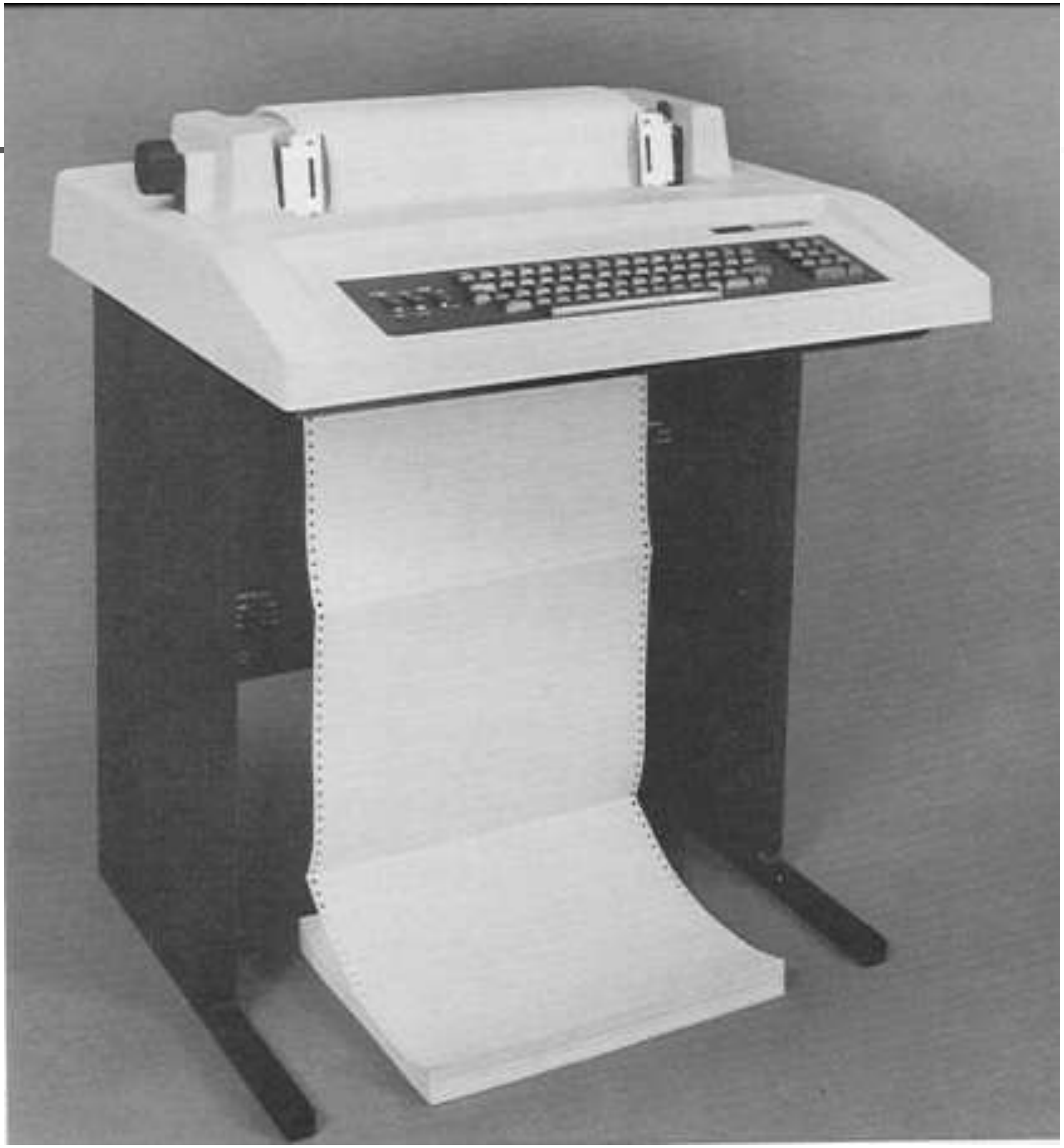
Developing large systems

11 thousand pages of continuous printer paper (used before laser printers)

As tall as the woman in the picture, about 5 feet 4 inches high (165 cm)!



A continuous paper printer



Developing large systems

What is a large software system?

- Imagine a system where you print your code and it’s ***11k pages*** (this is about half a million lines of code)!
- Then, think of ***3 million*** lines of code: Space shuttle control software

Developing large systems

What is a large software system?

- Imagine a system where you print your code and it's ***11k pages*** (this is about half a million lines of code)!
- Then, think of ***3 million*** lines of code: Space shuttle control software
- Also, such a system must be really robust!
- And so, it is difficult to create!

Engineering large systems

- Everyone knows about the Apollo's Moon landings (Neil Armstrong was the first man to walk on the Moon on July 20, 1969)
- The Lunar Module and the Command/Service Module were controlled by software systems, even back then!
- In fact, the source code of the control software for the Apollo missions has been released to the public just recently!

  http://www.inverse.com/article/18084-apollo-11-guidance-source-code-software-github

Developing large systems

Margaret Hamilton, standing right next to the printed source code of the Apollo 11 guidance software system.

About 420,000 SLOC* of AGC code (Apollo Guidance Computer)

January 1, 1969

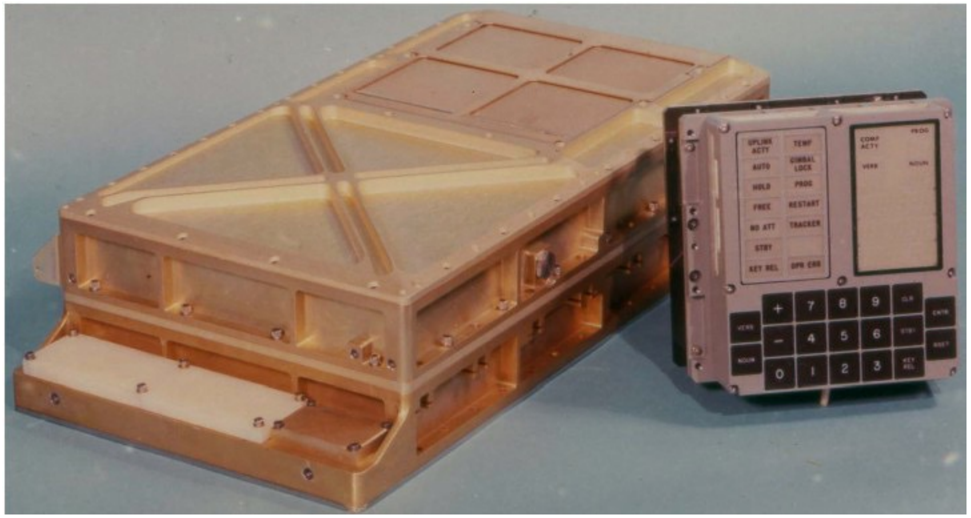
* Source Lines Of Code

Engineering large systems

- Margaret Hamilton directed a team which designed and developed the Apollo 11 on-board flight software running on Apollo Guidance Computer (AGC)
- She also coded much of the software herself

  http://www.ibiblio.org/apollo/

  https://github.com/chrislgarry/Apollo-11



Apollo Guidance Computer had 2048 words of RAM, 38,912 words of ROM with 15-bit words

wikipedia.org

Engineering large systems

- Margaret Hamilton was the director of the Software Engineering Division at the MIT Instrumentation Laboratory, contracted to develop the software for AGC
- She created a methodology of developing **mission-critical software systems**
- She is also credited with naming the discipline of creating large, complex, and high-quality software systems:

Engineering large systems

- Margaret Hamilton was the director of the Software Engineering Division at the MIT Instrumentation Laboratory, contracted to develop the software for AGC
- She created a methodology of developing **mission-critical software systems**
- She is also credited with naming the discipline of creating large, complex, and high-quality software systems:

  Software Engineering

- She later directed the Skylab software design and development.

Engineering large systems

Margaret Hamilton
is being awarded
The Presidential Medal
of Freedom

November 22, 2016

Engineering large systems

Some perspective on the “software size” [1]

SLOC	System
6.5 million	787 Dreamliner onboard systems
12 million	Android operating system
15.8 million	Linux Kernel (3.10)
40 million	Windows 7
86 million	Mac OS X 10.4 (Tiger)
150 million	2016 Ford F150 on-truck software
2 billion	All of Google services software

1. Based on various Web-accessible sources

Engineering large systems

Facebook users:

The system has about

62 million lines of code,

including all its systems

Software Development Process

Methodology of developing software systems is frequently referred to as the Software Development Process, or simply the Software Process

Software process is a structured set of activities required to develop a software system

Software Development Process

Many different software processes exist, but majority involve:

- Specification – defining what the system should do
- Design and implementation – defining the organization of the system and implementing the system
- Verification and Validation – checking that it correctly does what the customer wants
- Evolution – modifying the system in response to changing customer/user needs

Types of Software Processes

Plan-driven processes

all of the process activities are planned in advance and progress is measured against this plan

Agile processes

activities are incremental and iterative, and it is easier to change the process to reflect changing customer requirements

Types of Software Processes

- Most practical processes include elements of both the plan-driven and agile approaches
- There are no *right* or *wrong* software processes

Software Process Models

- Waterfall model
- Incremental development
- Iterative development
- Agile development
- Reuse-oriented development
- Prototyping
- Spiral model
- Formal transformation

Software Cost Distribution

Estimated cost	Phase
2%	Concept & Definition
4%	Requirements Definition
7%	Software Architecture Design
6%	Detailed Software Design
7%	Code and Unit Test
12%	Integration and System Test
3%	Acceptance Testing
1%	Replication, Storage and Shipment
2%	Delivery, installation and Training
55%	Maintenance
1%	Retirement

Source: http://www.12207.com/live-cycle-cost.html

Agile Development

- In the late 90s, Agile Development has been created as a response to fully planned, “heavy-weight” development.
- Created to address constantly changing requirements and need for rapid software development.
- Experience shows that when heavy-weight processes are applied to small/medium systems, the overhead of planning and design dominates the development process.

Agile Development

- System is developed through small, frequent, incremental releases.
- Requirements include relatively simple customer/user stories, easy to modify.
- Complete requirements document is usually not created.
- Customers are continuously engaged, and their representatives often take part in the development.

Agile Development

Agile methodologies include:

- Extreme Programming (XP)
- Dynamic systems development method (DSDM)
- Kanban (inspired by Toyota Production System and lean manufacturing)
- Scrum

Scrum Methodology

First introduced to software development by J.V. Sutherland and K. Schwaber in 1995.

Scrum method participants (roles) include:

- Product owner
- Development team
- Scrum master

Product requirements are referred to as product backlog, and include features, bug fixes, non-functional requirements, etc.

Requirements are represented as user stories.

User Stories

- **User story**
  - Often used in Agile processes, e.g., Scrum
  - A short, simple description of a feature told from the perspective of the person who desires the new capability, usually a user or customer of the system.
  - Focuses on what the user wants to achieve, not what the user wants the system to do.
  - A set of user stories defines system requirements
  - Often recorded on a flash card or a post-it note

User Stories

- Usually, a user story follows a simple template:

  As a <type of user>,

  I want <to perform some task>

  so that I can <achieve some goal/benefit/value>.

- For example:

  As a user, I want to sign-in to the site from a login page so that I can use the system's services.

  As a student, I want to purchase a parking permit so that I can drive to school and park there.

User Stories

As a user, I want to be able to manage ads so that I can remove expired and erroneous ads.

As an academic advisor, I want to have filtering option of student transcripts.

As a bank account owner, I want to check my balance online so that I can keep track of my money 24 hours a day.

User Stories

As a user, I want to search for a doctor by specialty.

As a shopper, I want to view a list of products so I can select some to purchase.

As an online shopper, I want to check out so I can get my products shipped to me.

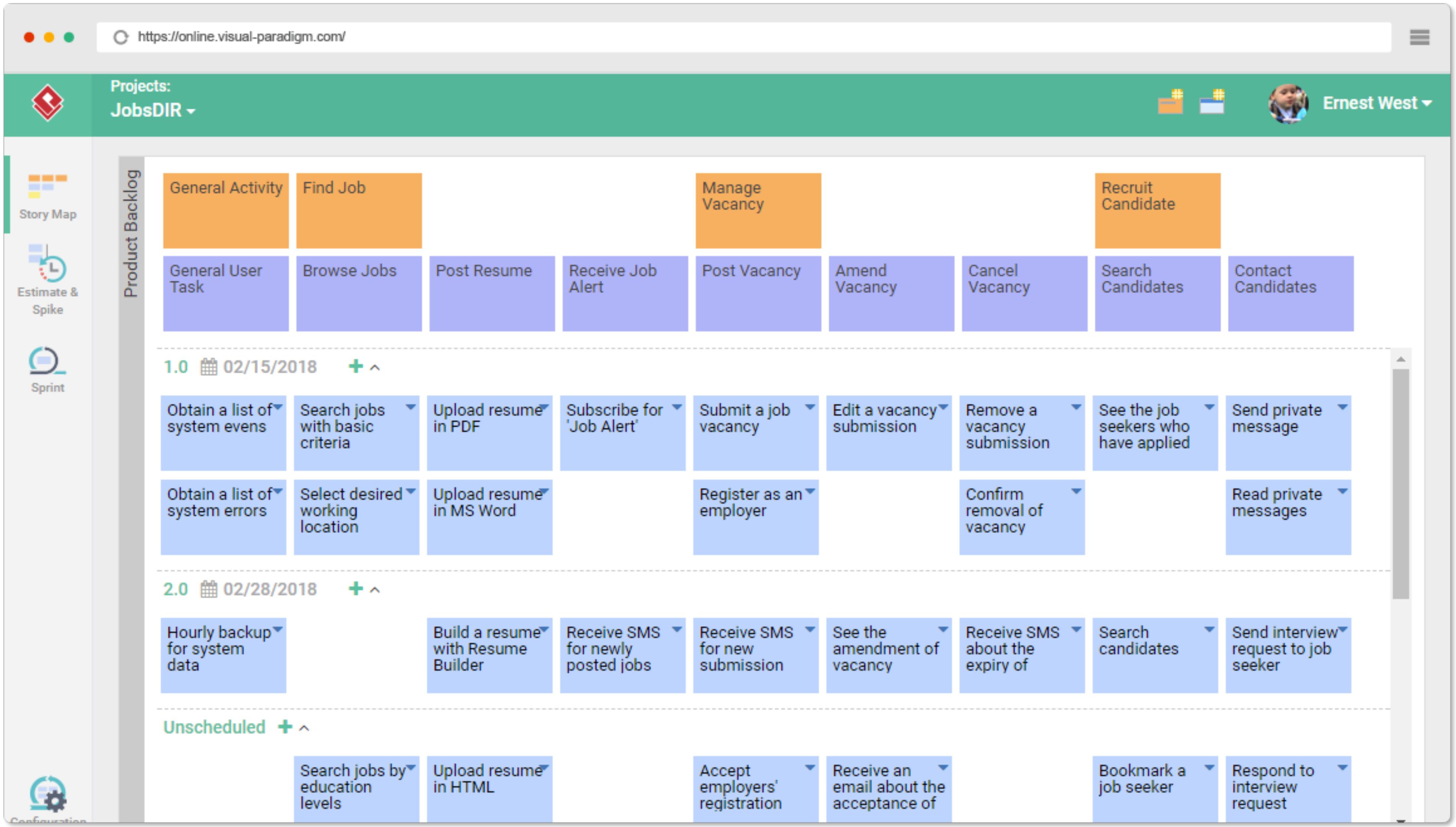
As a roommate in a store, I want to see the shopping list on my phone so I can purchase some or all the items on the list.

User Stories

As a user I want to be able to distinguish between bookmarks that relate to specific series/episode so I can browse easily.
As an editor I want to add bookmarks
As a user I want to explore
As a user I want help tips
Wireframes, UX + Journeys required

User stories on post-it notes

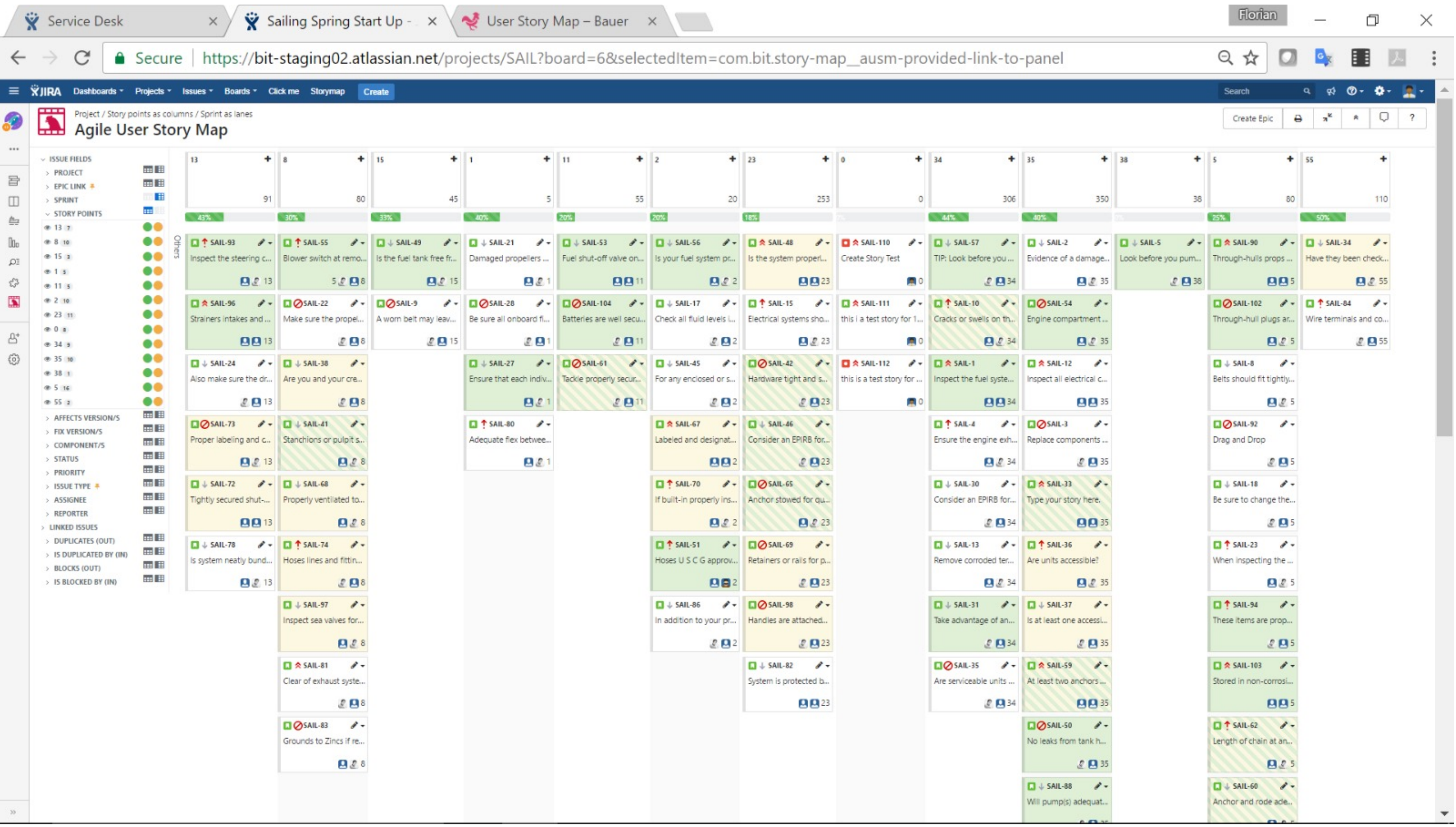
From: thoughtworks.com



General Activity	Find Job			Manage Vacancy			Recruit Candidate	
General User Task	Browse Jobs	Post Resume	Receive Job Alert	Post Vacancy	Amend Vacancy	Cancel Vacancy	Search Candidates	Contact Candidates
1.0 02/15/2018								
Obtain a list of system evens	Search jobs with basic criteria	Upload resume in PDF	Subscribe for 'Job Alert'	Submit a job vacancy	Edit a vacancy submission	Remove a vacancy submission	See the job seekers who have applied	Send private message
Obtain a list of system errors	Select desired working location	Upload resume in MS Word		Register as an employer		Confirm removal of vacancy		Read private messages
2.0 02/28/2018								
Hourly backup for system data		Build a resume with Resume Builder	Receive SMS for newly posted jobs	Receive SMS for new submission	See the amendment of vacancy	Receive SMS about the expiry of	Search candidates	Send interview request to job seeker
Unscheduled								
	Search jobs by education levels	Upload resume in HTML		Accept employers' registration	Receive an email about the acceptance of		Bookmark a job seeker	Respond to interview request

A collection of user stories in a software tool

Figure from: visual-paradigm.com



Tool: Agile User Story Map PRO for Jira

Figure from: atlassian.com

User Stories

Three C's of user stories:

- **Card**: The main intention is to describe the user story in short form to allow *common understanding* of the user need among all stakeholders
- **Conversation**: User stories shift the focus from writing about features to *discussing* them. In fact, these discussions are more important than whatever text is written
- **Confirmation**: *Acceptance tests* confirm that the story was delivered correctly

https://www.scrumalliance.org/community/articles/2013/september/agile-user-stories

User Stories

- Usually, a user story is accompanied by confirmation criteria (acceptance tests):

  As a customer, I want to withdraw cash from an ATM so that I don't have to wait in line at a bank office.

  Acceptance Criterion 1:

  ***Given** that the account is creditworthy*
  and the card is valid
  and the dispenser contains cash,
  ***When** the customer requests the cash*
  ***Then** ensure the account is debited*
  and ensure cash is dispensed
  and ensure the card is returned.

User Stories

As a customer, I want to withdraw cash from an ATM so that I don't have to wait in line at a bank office.

Acceptance Criterion 2:

***Given** that the account is overdrawn*

and the card is valid,

***When** the customer requests the cash*

***Then** ensure the rejection message is displayed*

and ensure cash is not dispensed.

User Stories

As a customer, I want to withdraw cash from an ATM so that I don't have to wait in line at a bank office.

Acceptance Criterion 2:

***Given** that the account is overdrawn*

and the card is valid,

***When** the customer requests the cash*

***Then** ensure the rejection message is displayed*

and ensure cash is not dispensed.

Acceptance criteria do not have to follow a specific "format". However, their intent should be clearly defined.

User Stories

As a user, I want to sign-in to the site from a login page so that I can be authenticated and use the site's services.

Acceptance Criteria:

Success

- *I can enter my email address and password and submit it for authentication.*
- *"Remember me" checked – store cookie for automatic login next time*
- *"Remember me" not checked – require login next time*

Failure

- *Illegal email format*
- *Unknown email/password*
- *System down for maintenance*

User Stories

As a conference attendee, I want to be able to register online, so I can register quickly and cut down on paperwork.

Acceptance Criteria:

- *A user cannot submit a form without completing all the mandatory fields.*
- *Information from the form is stored in the registrations database.*
- *Protection against robots is working (captcha).*
- *Payment can be made via credit card.*
- *An acknowledgment email is sent to the user after submitting the form.*

User Stories

- Some user stories can be large in scope and complex. These are called epics.
- Typically, an epic cannot be completed in a single sprint.
- Epics are split into smaller, regular user stories, after a conversation.
- Often, several related user stories are grouped together to form a theme, but a distinction epic vs. theme is sometimes unclear.

User Stories

Epic example:

As a hotel operator, I want to set the optimal rate for rooms in my hotel.

a. *As a hotel operator, I want to set the optimal rate for rooms based on prior year pricing.*

b. *As a hotel operator, I want to set the optimal rate for rooms based on what hotels comparable to mine are charging.*

c. *As a hotel operator, I want to set the optimal rate for rooms based on current projected occupancy.*

User Stories

As a user, I want to sign-in to the site from a login page so that I can be authenticated.

As an epic, it can lead to additional user stories:

As a new user, I want to register by creating a username and password so that the system can remember me and recognize me later.

As a registered user, I can log in with my username and password so I can trust the system.

As a registered user, I can change my password so that I can keep it secure or make it easier to remember.

User Stories

As a user, I want to sign-in to the site from a login page so that I can be authenticated.

As an epic, it can lead to additional user stories:

As a registered user, I want the system to warn me if my password is easy to guess so that my account is harder to break into.

As a forgetful user, I want to be able to reset my password.

As a registered user, I am notified if there have been three consecutive failed attempts to access my account, so that I am aware if someone is trying to access my account.

Scrum Methodology

Scrum workflow includes:

- Sprint planning – selection of product backlog items to be done
- Sprint – work on the sprint backlog
- Daily scrum – a daily review meeting
- Sprint review – the team demonstrates what they accomplished during the sprint
- Sprint retrospective – the team reflects on how they are doing and looks for ways to improve

Scrum Methodology

Scrum workflow

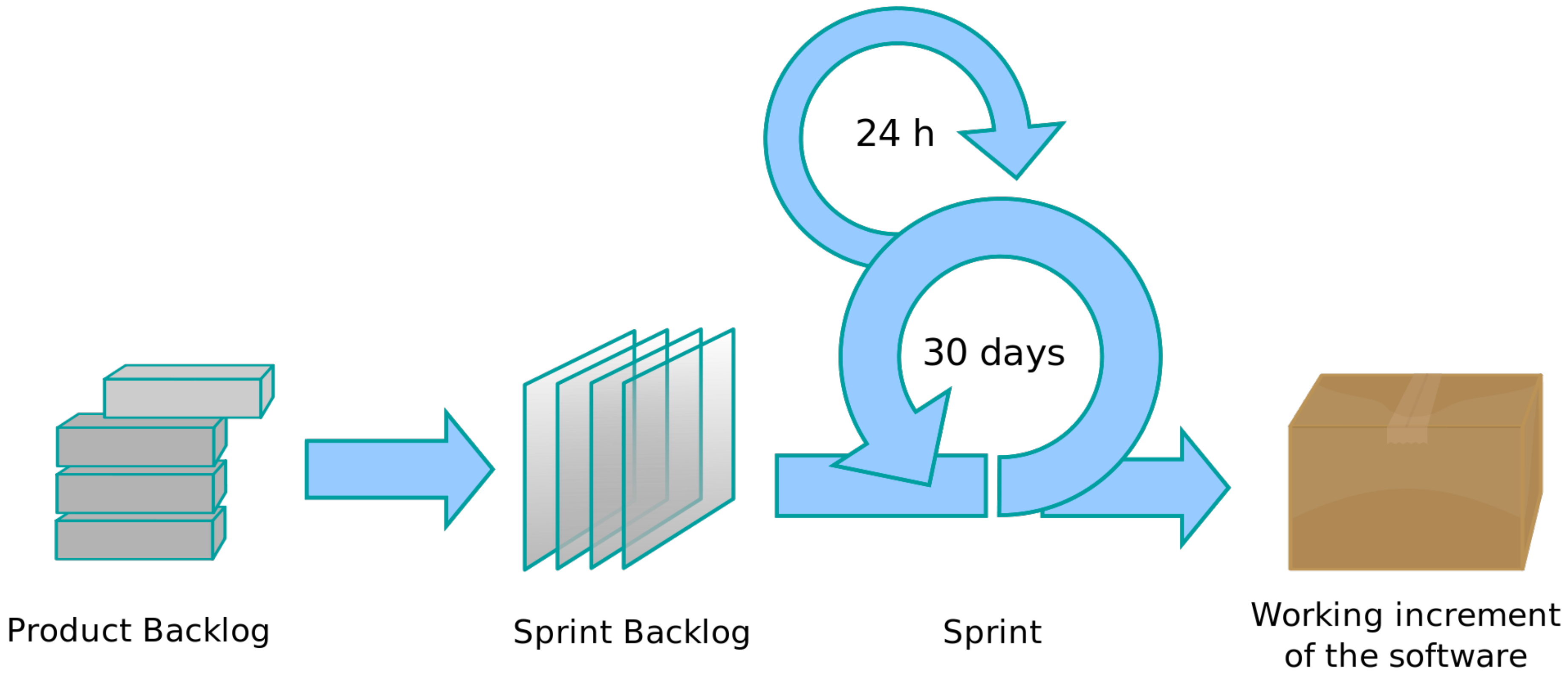


Figure from: wikipedia.org

A note on Agile development

- Agile development gained considerable popularity in the software industry
- It is regarded to be working well in small to medium size projects and especially for new software development
- Promotes close interaction with the users and customer

A note on Agile development

- Agile development is preferred for larger, multi-person team development of mobile applications
- A typical size of a larger mobile project is approximately 50k lines of code
- Often, much of the work is done by various frameworks, APIs, and other services

A note on Agile development

- Agile development is *not* considered as appropriate for safety-critical control systems (e.g., aircraft control), where complete system analysis is essential
- Not appropriate for very large systems
- Does not fit well when a software development contract must be drawn between large companies