ITA

Web Engineering: A Practitioner's Approach

by Roger S. Pressman and David Lowe

Chapter 1

- *Web-Based Systems*

WebApps

- The term *Web application* (WebApp) encompasses:
  - Everything from **a simple Web page**
    - Might be help consumer to compute an automobile lease payment
    - to **a comprehensive website** that provides complete travel services for business people and vacationers.
- Category:
  - Complete websites
  - Specialized functionality within websites
  - Information-processing applications that **reside on the Internet or on an Intranet or Extranet**

WebApps

- Means HTML, Java, XML, or any of the countless technologies that must be understood to build successful Web-based systems/applications (WebApps)
- WebApps can be pivotal to the success of all businesses and organizations

Web-Based Systems

- In the early days, the Web systems built using informality, urgency, intuition, and art
  - *Informality* leads to an easy work environment—one in which you can do your own thing.
  - *Urgency* leads to action and rapid decision making.
  - *Intuition* is an intangible quality that enables you to "feel" your way through complex situations.
  - *Art* leads to aesthetic form and function—to something that pleases those who encounter it.
- Problem is—this approach can and often does lead to problems

Web-Based Systems

- As WebApps become larger and more complex,
  - Informality remains, but some degree of requirements gathering and planning are necessary
  - Urgency remains, but it must be tempered by a recognition that decisions may have broad consequences
  - Intuition remains, but it must be augmented by proven management and technical patterns
  - Art remains, but it must be complemented with solid design
- Bottom line—we must adapt the old-school approach to the realities of a Web 2.0 world….and now Web 3.0 world

WebApp Attributes

- Data driven
- Performance – Not to wait too long for serverside processing, for client-side formatting and display
- Continuous evolution
- Immediacy - exhibit a time-to-market
- Network intensiveness - diverse community of clients on net
- Concurrency - Large number of users may access at one time
- Unpredictable load- No. of users of may vary from day to day.
- Availability
- Content sensitive- simple, yet meaningful for nontechnical user
- Security
- Aesthetics- appeal of a WebApp's look and feel

WebApp Types

- Informational- readonly content with simple navigation and links
- Download - informational and *download capability*
- Customizable – different for each different user
- Interaction – chat room
- User input – take input from user in form for automization
- Transaction-oriented – automated based on user request
- Service-oriented
- Portals - providing website links having answers for customer
- Database access
- Data warehousing

(see http://digitalenterprise.org/models/models.html for examples)

Web Apps

- Why Web Applications/Web based systems fail?
- Because many built in an ad hoc manner
  - With little regard to the
    - Fundamental principles of problem analysis
    - Effective design
    - Solid testing
    - Change management

And What's the Solution?

Web Engineering

Web Engineering

- Goal is to build WebApps or Web based system that **satisfy users' needs** and provide **real benefit** to their clients' businesses or organizations.
- *i.e. To build **industry-quality** WebApps*
- *An **agile**, yet **disciplined framework** for building **industry-quality** WebApps*

Agile Approach

- Business strategies and rules change rapidly
- Management demands near-instantaneous responsiveness (even when such **demands are completely unreasonable**)
- Stakeholders often don't understand the consequences of the Web and **keep changing their mind** even as they **demand rapid delivery**

An agile approach helps to manage with this **fluidity** and **uncertainty**

Agile Approach

- Able to appropriately respond to changes, Change is to
  - The software being built
  - The team members
  - New technology of all kinds that may have an impact on the product they build or the project that creates the product
- Support for **changes** should be built-in everything we do in software
- An agile team recognizes that software is developed by i**ndividuals** working in teams and that the **skills** of these people, their **ability** to collaborate is at the core for the success of the project

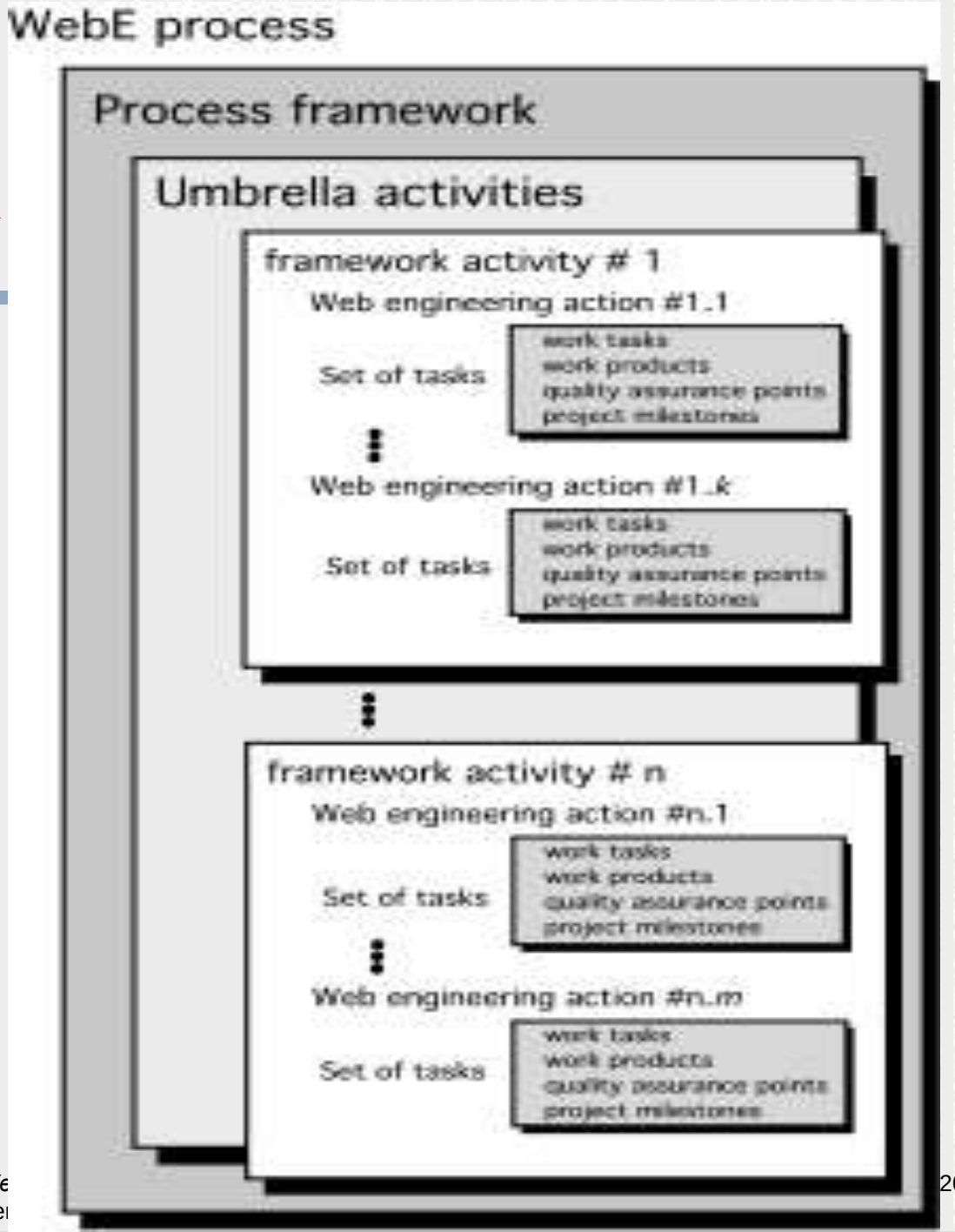
What is an Agile Process?

- Agile Web engineering combines a philosophy and a set of development guidelines. The philosophy encourages:
  - Customer satisfaction
  - Early incremental delivery of the WebApp
  - Small, highly motivated project teams
  - Informal methods
  - Minimal work products
  - Overall development simplicity
- An agile process stresses delivery over analysis and design and also **active and continuous communication** between developers and customers.

What is a WebE Framework?

- Framework
  - A **set of activities** that will *always* be performed for *every* Web Engineering project – though the nature of the activities might vary to suit the project
  - Each framework activity is composed of a **set of actions**
  - Actions encompass
    - **Work tasks**
    - **Work products**
    - **Quality assurance points**
    - **Project milestones**
  - A framework also has a set of "**umbrella activities**"

A Generic Framework



The WebE Framework: Activities

- **Communication**
- **Planning**
- **Modeling**
- **Construction**
- **Deployment**

The WebE Framework: Activities

- **Communication.** Involves heavy interaction and collaboration with the customer (and other stakeholders) and encompasses requirements gathering and other related activities.
- **Planning.** Establishes an incremental plan for the WebE work.
- **Modeling.** Encompasses the creation of models that assist the developer and the customer to better understand WebApp requirements and the design
- **Construction.** Combines both the generation of HTML, XML, Java, and similar code with testing that is required to uncover errors in the code.
- **Deployment.** Delivers a WebApp increment to the customer who evaluates it and provides feedback based on the evaluation.

Adapting the Framework

- Adapt
  - to the problem
  - to the project
  - to the team
  - to the organizational culture
  - to adapt throughout the project as circumstances **change**!

Adapting the Framework

- Adaptation leads to,
  - Overall flow of activities, actions, and tasks and the interdependencies among them.
  - Degree to which **work tasks are defined** within each framework activity.
  - Degree to which **work products are identified** and required.
  - Manner in which **quality assurance** activities are applied.
  - Manner in which **project tracking and control activities** are applied.
  - Overall **degree of detail** and consistency with which the process is described
  - Degree to which **customers and other stakeholders** are involved with the project
  - Level of **autonomy given to the software project team**
  - Degree to which **team organization and roles are prescribed**

Underlying Agility Principles

1. **Highest priority is to satisfy the customer** through early and continuous delivery of valuable software.
2. Welcome **changing requirements**, even late in development. Agile processes harness **continuous change for the customer's competitive advantage**.
3. **Deliver working software increments frequently**, from as often as every few days to every few months, with a preference to the shorter timescales.

Underlying Agility Principles

4. Business people and developers **must work together** daily throughout the project.
5. Build projects around motivated people. Give them needed **environment and support**, and trust them to get the job done.
6. The most efficient and effective method of **conveying information** to and within a development team is face-to-face conversation.

Underlying Agility Principles

7. **Working software** is the primary measure of progress.
8. Agile processes promote **sustainable development**. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to **technical excellence and good design** enhances agility.

Underlying Agility Principles

10. **Simplicity**—the art of maximizing the amount of work not done—is essential.
11. The best architectures, requirements, and designs emerge from **self-organizing teams**.
12. At **regular intervals, the team reflects** on how to become more effective, then tunes and adjusts its behavior accordingly.

Web Engineering = Software Engineering ?

- Software engineering principles, concepts, and methods can be applied to Web development, but their application **requires a somewhat different approach** than their use during the development of conventional software based systems.
- Software engineering is a layered technology

Software Engineering Layers

- Quality: Foster a continuous process improvement culture
- Process: The glue that holds the technology layers together
  - Work products (e.g., models and documents) are produced, milestones are established, quality is ensured, and change is properly managed
- Methods: Provide the technical how-to's
  - Communication, requirements analysis, design modeling, program construction, testing, and support.
- Tools: Support for the process and the methods

Web Engineering

- Web Engineering differed from Software Engineering…
  - WebE framework must be defined within a process that:

    (1) embraces change,

    (2) encourages the creativity and independence of development staff and strong interaction with WebApp stakeholders,

    (3) builds systems using small development teams, and

    (4) emphasizes incremental development using short development cycles

WebE Methods

- Encompasses a set of technical tasks that enable a Web engineer to understand, characterize, and then build a high-quality WebApp
  1. Communication methods
  2. Requirements analysis methods
  3. Design methods
  4. Construction methods
  5. Testing methods

WebE Methods

1. Communication methods
   - Define the approach used to facilitate communication between Web engineers and all other **WebApp stakeholders** (e.g., end users, business clients, problem domain experts, content designers, team leaders, project managers)
   - Communication techniques are important during requirements gathering and whenever a WebApp increment is to be evaluated
2. Requirements analysis methods
   - Provides understanding the deliverable content of a WebApp, functions for the end user, and the navigation modes of interaction for each class of user

WebE Methods

3. Design methods
   - Design techniques for WebApp content, application and information architecture, interface design, and navigation structure
4. Construction methods
   - Set of languages, tools, and related technology to create WebApp
5. Testing methods
   - Testing component-level and architectural issues
   - Navigation testing
   - Usability testing
   - Security testing
   - Configuration testing

WebE Methods

- Other than these are
  - Project management techniques
    - Estimation
    - Scheduling
    - Risk analysis
  - Software configuration management techniques
  - Review techniques

Industry-Quality WebApps

Characteristics

- Take the time to understand business needs and product objectives, even if the details of the WebApp are vague.
- Describe how users will interact with the WebApp using a scenario-based approach.
- *Always d*evelop a project plan, even if it's very brief.
- Spend some time modeling what it is that you're going to build.
- Review the models for consistency and quality.
- Use tools and technology that enable you to construct the system with as many reusable components as possible.
- Don't reinvent when you can reuse.
- Don't rely on early users to debug the WebApp—design and use comprehensive tests before releasing the system.

Industry-Quality WebApps

Characteristics :

1. Take the time to understand business needs and product objectives, even if the details of the WebApp are vague
   - Many WebApp developers erroneously believe that vague requirements (**which are quite common**) relieve them from the need to be sure that the system they are about to engineer has a **legitimate business purpose.**
   - The end result is (too often) good technical work that results in the **wrong system being built for the wrong reasons and for the wrong audience**
   - If stakeholders cannot describe a business need for the WebApp, **proceed with extreme caution**
   - If stakeholders struggle to identify **a set of clear objectives** for the product (WebApp), do not proceed until they can

Industry-Quality WebApps

Characteristics :

2. Describe how users will interact with the WebApp **using a scenario based approach**
   - Stakeholders should be convinced to develop scenarios that reflect how various users will interact with the WebApp
   - These scenarios can then be used:

     (1) for project **planning and tracking**,

     (2) to guide analysis and **design modeling**, and

     (3) as important input for the design of tests

Industry-Quality WebApps

Characteristics :

3. Develop a **project plan**, even if it's very brief
   - Base the plan on a process framework that is **acceptable** to all stakeholders
   - Because project time lines are very short, use a **"fine" granularity** for schedule-project should be **scheduled and tracked** on a daily basis
   - Many WebApp developers erroneously believe that vague requirements (quite common) relieve them from the need to be sure that the system they are about to engineer has a legitimate business purpose
     - The end result is (too often) good technical work that results in the wrong system being built for the wrong reasons and for the wrong audience
     - If stakeholders cannot describe a business need for the WebApp, proceed with extreme caution

Industry-Quality WebApps

Characteristics :

4. Spend some time **modeling** what it is that you're going to build
   - Generally, comprehensive analysis and design documentation is ***not developed*** as a part of Web engineering work
   - However**, well-targeted graphical models** can and do illuminate important engineering issues

Industry-Quality WebApps

Characteristics :

5. Review the models **for consistency and quality**
   - Pair walkthroughs and other types of reviews should be conducted throughout a WebE project
   - The time spent on reviews pays important dividends because it often **eliminates rework and results in a high-quality WebApp**—thereby increasing customer satisfaction

Industry-Quality WebApps

Characteristics :

6. Use **tools and technology** that enable you to construct the system with as many **reusable components** as possible
   - A wide array of WebApp tools is available for virtually every aspect of the WebApp construction
   - Many of these tools enable a Web engineer to build significant portions of the application using reusable components

Industry-Quality WebApps

Characteristics :

7. Don’t **reinvent** when you can reuse
   - A wide range of design patterns have been developed for WebApps
   - These patterns allow a WebE team to develop architectural, navigation, and component-level details quickly using proven templates

Industry-Quality WebApps

Characteristics :

8. **Don't rely** on early users to debug the WebApp—design comprehensive tests and execute them before releasing the system
   - Users of a WebApp will often give it one chance. If it fails to perform, they move elsewhere—never to return
   - It is for this reason that "test first, then deploy" should be an overriding philosophy, even if deadlines must be stretched.

The WebE Process

- The process must be **agile and adaptable**, but it must also be *incremental*
- Why incremental?
  - Requirements evolve over time
  - Changes will occur frequently (and always at inconvenient times
  - Time lines are short
- Incremental delivery allows you to manage this change!

Incremental Delivery

Repeat the development cycle for each increment!

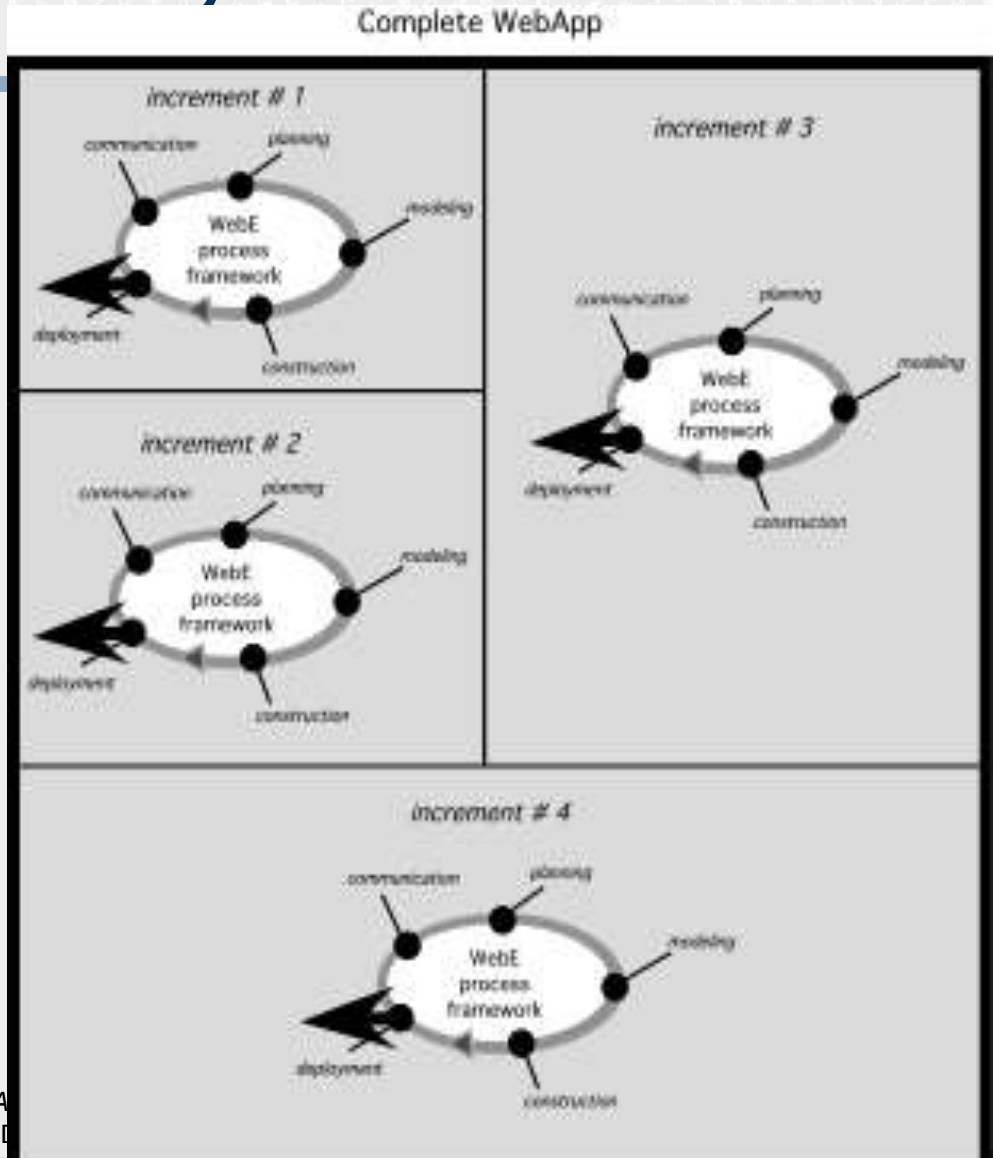
-Communication

-Planning

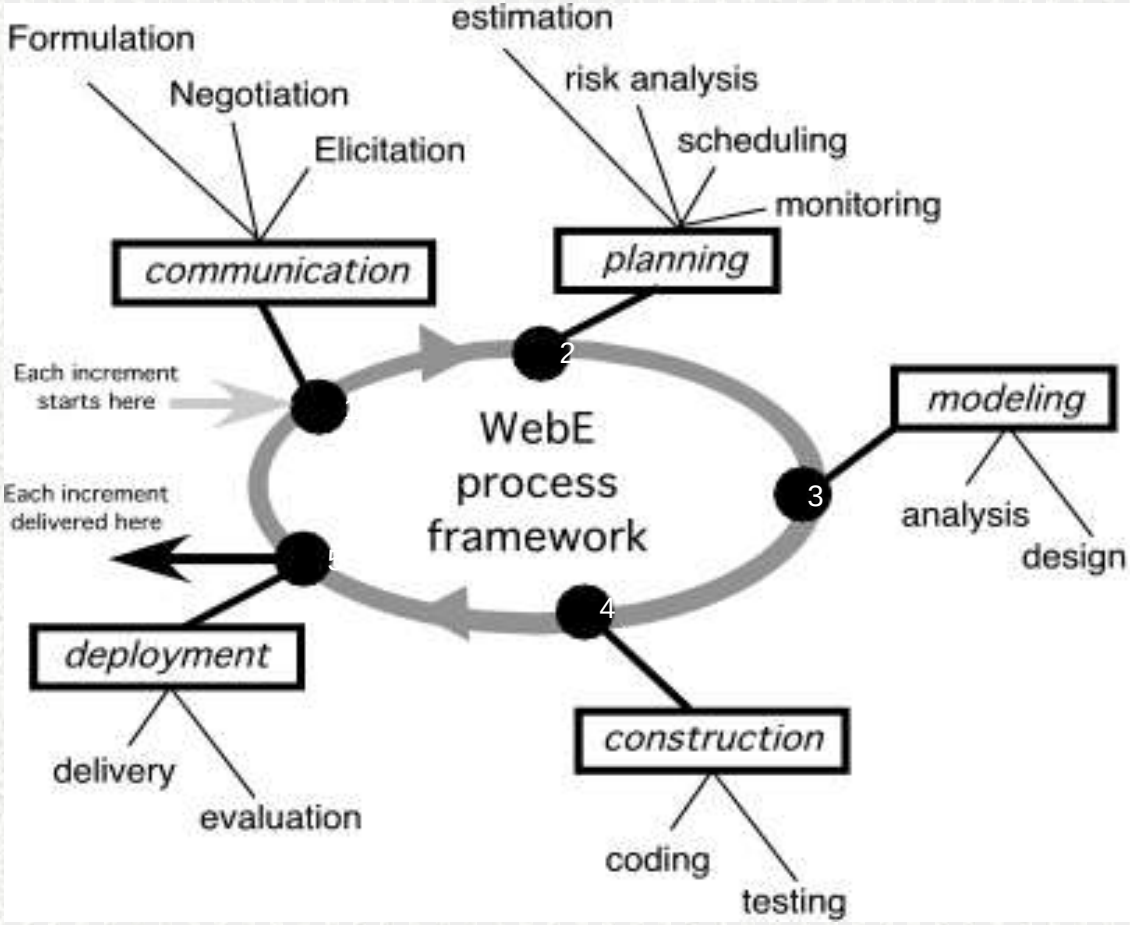
-Modeling

-Construction

-Deployment



WebE Process Activities & Actions



WebE Process: Communication

- *Communication is the activity that establishes the "destination" for a WebApp* project
- For a simple destination, there are a relatively small number of informal actions and tasks required to be sure you know where you're going
- If the destination is more difficult to describe, you'll need to refine the communication activity with more care

WebE Process: Communication

- Objectives:
  - Identify the
    - business stakeholders
    - user categories
    - the problem to Solve.
    - Input/Output for the End User.
    - functionality required to manipulate data
  - Formulate the business context
  - Develop usage scenarios
  - Gather requirements about content, interaction metaphor, Computational functions, navigation schema

- Define the
  - key **business goals** and objectives for the WebApp
  - informational and applicative goals for classes of content are to be provided to end users
  - how **dynamic** is the content; that is, how often does it change…

WebE Process: Planning

- Activity requiring the following to model, construct, and deploy the increment
  - **Estimation** of effort and time required to deploy the increment
    - In terms of person-days and time (calendar days)
    - Resources (people, hardware, software) required to do the work
  - **Assessment** of risks associated with the delivery of the increment high-probability, high-impact risks to be mitigated
  - **Development** schedule for the increment for tasks to be allocated along the time line and to establish intermediate milestones
  - Establishment of
    - Work products (written scenarios, sketches, models, documents) to be produced as a consequence of each framework activity
    - Quality assurance approach

WebE Process: Modeling

- *An **activity*** that creates one or more conceptual representations of some aspect of the WebApp to be built
- A *conceptual representation-one or more of the following* forms:
  1. Written documents
  2. Sketches
  3. Schematic diagrams
  4. Graphical models
  5. Written scenarios
  6. Paper or executable prototypes
  7. Executable code
- Two Web engineering actions occur during modeling:
  - *Analysis*
  - *Design*

WebE Process: Analysis Modeling

- This model focuses on WebApp **content, modes of interaction (including navigation), functionality, and the technical configuration of the WebApp**
- From the communication,
  - If information exists and is complete, → **no need for analysis modeling** for the increment
  - If the information is **incomplete** or implies a degree of complexity → demands further examination, **proceed to the analysis modeling tasks**

WebE Process: Analysis Modeling

- The following tasks are to be done during an analysis model:
  - Represent WebApp content, decide **static** (do not change based on user type/input) and **dynamic** (generated based on user type/input)
  - Identify **content** form and style of each content class and relationships among them
  - Refine and extend **user scenarios for input**, steps and to check consistency and of enough detail
  - Create an **interaction model for complex** scenarios showing the relationship between user and each task and user actions transition from one state to another
  - Refine interface requirements for **menus, layout, navigation**
  - Identify **functions and user input** for function
  - **Define constraints and performance requirements** and privacy policy
  - Identify database requirements, content interface for the database

WebE Process: Design Modeling

- The goal is to produce a **model or representation**
- If the increment is well understood and very easy to construct, the only design model must be a simple sketch
- If, on the other hand, the increment is more complex, a **more detailed design model** may be created

WebE Process: Design Modeling

- The model can consider some/all aspects of WebApp design:
  - Interface design
  - Aesthetic design
  - Content design
  - Navigation design
  - Architecture design
  - Component design

WebE Process: Design Modeling

- The model can consider some/all aspects of WebApp design:
  - Interface design
    - Describes the structure and organization of the user interface
    - Includes a representation of screen layout, a definition of the modes of interaction, and a description of navigation mechanisms
  - Aesthetic design
    - E.g. graphic design, describes the “look and feel” of the WebApp
    - Includes color schemes, geometric layout, text size, font and placement, the use of graphics, and related aesthetic decisions
  - Content design
    - Defines the layout, structure, and outline for all content
    - Establishes the relationships among content objects

WebE Process: Design Modeling

- The model can consider some/all aspects of WebApp design:
  - Navigation design
    - Represents the navigational flow among content objects and functions
  - Architecture design
    - Identifies the overall hypermedia structure for the WebApp
  - Component design
    - Develops the detailed processing logic required to implement functional components that implement a complete WebApp function

WebE Process: Design Modeling

- The following to consider to develop a design model:
  - For each **usage scenario**, design the
    - Interaction **tasks and subtasks** to be represented as part of the interface
    - Required **interface control mechanisms** (e.g., links, buttons, menus)
    - **Control mechanisms** positions on a Web page
  - Design the aesthetic for the WebApp user categories by considering consistent
    - Page layout , color and form, navigation mechanisms positions and representation, all logos, graphics, images, and backgrounds implemented
  - Design the content/databases and data structures required to implement functionality or to display content
  - Design appropriate security and privacy mechanisms

WebE Process: Construction

- As construction proceeds, you can perform two WebE actions:
  - Code generation
  - Testing
- The following tasks help you plan the code generation action:
  - Build and/or acquire all content, and integrate the content into the WebApp architecture
  - Select the appropriate tool set for the generation of HTML code
  - Implement each page layout, function, form, and navigation capability
  - Implement all forms, scripts, and database interfaces and computation functions for the client/server side
  - Address configuration issues of browsers, plug-ins, and operating system environments for both the client/server sides

WebE Process: Construction

- Once the WebApp has been constructed, it must be tested
- Testing begins with a relatively narrow focus and then continues to exercise a broader view of the WebApp
- The following tasks to plan the testing action :
  - Test all WebApp components (content and function)
  - Test navigation

WebE Process: Deploy

- The following tasks are considered to deploy the WebApp increment:
  - Deliver the WebApp increment to a server at **a predefined domain**
  - Establish an **online feedback mechanism** for end users
  - Evaluate **end-user interaction**, assess lessons learned and consider all end-user feedback and make modifications to the WebApp increment as required