eSpace Development Process

Document Version: 3.0

Date: 03/03/10

Document Versions

Version	Date	Modifier	changes
<1.0>	09/16/08	Heba Hosny	
<2.0>	07/19/09	Ali Maher	
<3.0>	03/03/10	Ali Maher	1-[sec II] New objectives (starting to focus on enabling the next effort) 2-[sec 2] Design questionnaire before contract writing. 3-[sec 3] a) considering vacation plan. b) calculating velocity. 4-[sec 5] Bromine as test script management tool. 5-[sec 6] Detailed SVN structure. 6-[sec 10] Point (5) dedicating time for documentation. Point (e) updating team velocity. 7-[sec 11] **Newly added section:** Closing Iteration. 8-[sec 12] Celebrating the sign-off. 9-[sec 14] **Newly added section:** Testing. 10-[sec 15] Point (5) Project Template, (6) Standard code style. 11-[sec 16] **Newly added section:** Enabling next effort. 12-[sec 17] More value oriented measurements. 13-[sec 18] Tech talk as an education channel. 14-[sec 19] Point (5) Test scripts as dynamic documentation, Point (6) Handling customer additional documentation requirements. 15-[sec 20] Point (e) mocking external systems, Point (g) business handover.

I-Introduction

This document describes eSpace Development Process (Support, Research, Consultation, … have other process descriptions). This description includes:
a- Development Team.
b- Development practices.
c- Development work flow.
d- Communication with the other teams.
c- Process management tools.

II-Objectives and Values

1.Customer's satisfaction: deliver the right product on time and on budget with high quality.

2.Keeping eSpace spirit and environment.

3.Minimizing process overhead on both Tech and Managerial levels.

4.Enabling the next effort by (Calculating Velocity, Collecting measurements, Agile Documentation.)

5.Smoothing the work flow and reducing the chances of problems.

6.Preparation for expansion.

1-Team and Roles

Team must not be less than **4** members; a **UI/Graphic designer**, a **tester** and 2 **developers** one of them is a **team leader**. Note that there are some other roles that are important for the success of the project but they are not fully dedicated to the project and not under the control of the team leader. These roles are: **Product Owner (still a proposal)**, **Solutions admin**.

2-Contract writing

1- Customer send his description of the required system in a form of requirement document.

2- Operations manager assign the best team to the new project.

3- Operations manager sends the requirement document to the team leader.

4- Team leader converts this document into a set of user stories covering the scope but not the details of the project.

5-Design team **meet with the client, and if not possible,** communicate with the client via sending a design questionnaire to collect the vision, marketing, graphics and design aspects. Client must answer this questionnaire before
contract signing.

6- Team leader gives the “size” estimate for each story in story points (or ideal days). See “estimation” section below. Then he passes these estimates back to the operations manager.

7- This estimation must be reflected in contract in the following articles:

a) Deliverables: Stated in the contract in the form of themes. **Theme** is the deliverable of a set of consecutive iterations that add a new value to customer's business.

b) Schedule (time line): Stated in the contract as Release plan. Expected calender date for each theme. Taking vacation plans into consideration is very important here.

c) Changes: Contract must clearly state that we only accept changes within the Deliverables scope.

d) Communication: Contract must clearly state a limited review and feedback period for the customer after each delivery.

e) Payments: Every theme should have its corresponding payment. Theme may be equivalent to one iteration only based on the nature of the project.

3-Estimation

1-Every Story has 2 estimations. One for development and another for UI design.
2-Testing (QC) is estimated to 3 days per 2 weeks iteration for manual and automated test.
3-Team leader and operation manager can decide to only use development estimation and put one full time designer for the project development life time.
4-Team leader is the responsible and committed to the estimates.
5-Team leader is highly encouraged to share his estimates with (part of) his team.
6-Estimates must cover all the time needed in the ideal cases. No time needed for hidden or extra tasks.
7-Scheduling: Operation manager and team leader are responsible for translating the estimates into Calender release plan after buffering the estimates with corresponding risks. Vacation plan must be put into consideration. Other stuff like traveling , accommodation,... must be considered too.
8-Using team velocity and story history improves the estimation and scheduling.

4-Project kickoff meeting

Kickoff meeting is the first gathering for <u>all team members</u> and the operations manager. The output of this meeting is:

1. **Team** is assigned to the project.
2. **Scope** (list of project features) is determined.
3. **Goals** (examples: customer trust, building experience in new tech, ...) are stated clearly.
4. **Final product** is described.
5. **Investigations , learnings and risks** are determined.
6. **Release plan** is done. (Iteration's calender deadlines (start, end, estimated story point, derived estimated man-days. Note that man-day is a normal business day for single developer. He is expected to pay 8 hours per day which will result in 6.8 hours net), cost and resources are defined.
7. **Themes** are stated (as written in the contract).
8. The start and end day (of the week) for the project iterations and testing days (see Testing below).

Important meeting points <u>must</u> be documented in the *Wiki* of the project

5-Project Setup

All tools are setup and working on servers and developers' machines.
Tools are for:

1. A revision control system tool, *i.e. SVN* .
2. A project management tool, *i.e. Redmine.*
3. A UI automatic testing tool, *i.e. Selenium, Bromine.*
4. An IDE with all Plugins, *i.e. eclipse,.*
5. A web UI for managing deployments, *i.e. Webistrano.*
6. A testing environment server; *staging server.*
7. An integration testing tool, i.e. a recipe for testing integration in *Webistrano.*
8. A code coverage test tool, *i.e. rcov.*
9. Defects tracking tool, if it's not embedded in the project management tool.

6-SVN Structure

Any project at eSpace (service, product, library, component,...) must have the following SVN structure: <Project name>/trunk

/branches

/tags

-trunk contains directly the project version that is currently under development. It may be unstable (but must be running and passing the unit tests) or finished.

-Branches are used for project versions that may need major, risky, complex,.. changes.

-Each iteration (in release 1,2,.. or in support) has a deliverable at its end. A "tag" must be created with the following naming convention: <release>.<iteration>.<stability>.<build>

Release: The project release in which the code is developed. If the project is new then the team is working in release #1 from the first day.

Iteration: An increasing counter of the number of iterations.

Stability: even = stable , odd = unstable

Build: the release number of the SVN

If the deliverable is ready to deploy on the staging server, another tag called "testing" should be created. If it is ready for production this tag should be "stable".

7-Learning and Investigations

If the project requires investigating new tools or technologies, or requires acquiring new knowledge, then this must be done as follows:

a) Estimate the time they will take in the investigation or learning.
b) State the output of the investigation or learning in terms of working software.
c) Add this to the project tasks.

8-Architecture Design

1) Designing the system **architecture** and **main modules**.
2) Defining the **reusable components** that can be reused (from open source projects or eSpace's previous projects) in this project.
3) Documenting the architecture on the Wiki.

9-Iteration 0

Iteration 0 is the first iteration in the project. It is predefined to be only one business week length. It should be taken into consideration when estimating and planning. It is dedicated for the following tasks:

1- Project setup (see point 4 above)

2- Learning and investigations (see point 5 above)

3- Architecture design (see point 6 above)

4- Technical pre-analysis

5- Domain pre-analysis and design documentation.

6- Story prioritizing and planning for Iteration 1.

10-Iteration Work flow

Iteration must start and finish on one of Mon, Tue, Wed. (any extra days should be in the first iteration)

1. Start a new **iteration** in the project management tool, and fill it with the **stories** that have the highest instant priority. Each story has to have (beside its previously estimated points):
    a) Title.
    b) Description.
    c) Time estimate.
    d) Due time.

This is done by a planning meeting with the customer or by sending him the final iteration plan to review. All stories details must be discussed and assured to be in the scope of estimation before going to the next step. If discussions results in out of scope details then see customer handling section below.

2. Assign each story to a member. The assigned member must divide the story into **tasks** and give each task an estimate.
3. Hold a wire frames meeting between the developers and the designer. Wire frames have to contain the main work flow design and privileges for each of the system users. The output must be accepted by the customer.
4. Beside the stories, iteration plan must include **defects fixing** for pending defects based on their **severity.** Iteration plan also may include any remaining tasks from a previous iteration.
5. In each Iteration add 0.5 team day to document the previous iteration. See documentation

section below.

For every story, tasks has to be done as follows:

1. Designers should deliver the interface design (HTML, CSS, Flash and effects JS in the case of building web applications). Developers then fill those interfaces with dynamic code. It's not allowed for developers to change those GUI files.
2. Developers then implement both main code and its required unit/functional tests individually or on pairs. Code must at least be covered by 80% by unit tests.
3. Do code Refactoring whenever you need. This will enhance the design by time.
4. Ensure that the code passes the unit tests.
5. A code review has to be done by any one in the team. It is not allowed to commit code without review.
6. **Commit** the code in the version control system, *SVN,* (based on the previous two steps this is an approved commit).
7. Record the actual time of your task.
8. Team leader has to run both integration tests and coverage tests to be sure of their pass, QA should run them as well.
9. **Build, test and deploy** with project deployment tool. Never to leave code with failure in building, testing or deployment. Note: Deployment will be on the *staging* server which may be accessed by the customer.
10. Tester has to write the GUI **functional** tests using UI automatic testing tool (*Selenium*) and add them to test script management tool (*Bromine*).
11. Task is finished if it is **accepted** by the team leader and the tester who's responsible of closing it before reviewing it by the customer. Note that the task life cycle is:

    new (created by developer) --> resolved (finished by developer) -->closed (accepted by tester) / reopened (rejected by tester or customer)

12. The task may not accepted. This may lead to one of the following states:
    a) *Severe*: task is not completed and developer must fix it immediately (before moving to the next task) because his task is not finished yet.
    b) *defects*: task is completed and defects will be opened in *defects tracking tool* (*Bugzilla*).
    c) *New feature*: task is completed and new Feature(s) is added as a task in *project management tool* (*Redmine*).
13. Introducing new tasks may lead to one of the following **decisions:** Refactoring, logic changes, integration issues, change layout design or workflow design, ... Any decision must be a team decision. This must be reflected in **Documentation** and **Tests** in the next iteration (if needed).
14. Iteration closing
    a) At the end of the iteration the planned deliverable should be tested, accepted and deployed on the staging server. A tag must be created at SVN as described in section 5.

b) If the iteration's due date is detained, new due date has to be specified in project management tool, and that must be reflected in the project plan. This update has to be announced early and clearly to all parties.

c) Iterations should be closed in the middle of the working week; it's not recommended that iterations be due for delivery on days that immediately precede official vacations.

d) Iteration must be closed. A decision may be taken to extend the due date of the iteration. This extension must not to exceed double the time estimated for it.

e) Update Team Velocity. Team Velocity = avg(New Iteration Velocity, Team Velocity)

   Where New Iteration Velocity = actual finished points (without reestimation) / (number of mandays per iteration). Its unit is point/manday for a specific team.

11-Closing iteration: (n+1) Iteration

Closing iteration is one week iteration follows all of the production Iterations. In this iteration no new story is allowed. The team will focus on:

1-Putting the project in production state

2-Fixing any remaining severe, major or normal bug.

3-Implementing feedback from the last production iteration.

4-Finishing, fine tuning and general testing of the complete work delivered.

5-Going live and be close to the customer to help in any operational problems, if part of original agreement. Not all agreements are bound to actually going live.

6-Giving the customer the sense of security and care.

7-Conducting Sign-Off meeting.

12-Project sign-off meeting

Signing off is an important step that helps in:

1.Clear Announcement of the closure of the current release
2.Starting support (if needed)
3.Collecting customer feedback
4.Writing down the lessons learned
5.Increase the moral of the team.
6.Generation of new ideas.

This may be done by sign off meeting with the customer and then by an internal small celebration inside eSpace.

13-Customer handling

Customer is free to change his requirements. We have to respond to his changes based on the nature of each change (Ordered by their difficulty):

1-Small changes: small change is the change that requires around one or two hours and will not affect our architecture or add new stories to the system. We should quickly accept these changes. This helps in building the trust.

2-Non-implemented details changes: Changes in the details of a non-implemented story. We should accept these changes as long as it will not change the old story into a totally new (more time consuming) one.

3-Implemented details changes: After implementing a story, the customer is asked to review it and give us his feedback. If he send his feedback within the period stated in the contract then it may be duo to one of the following reasons:

a) Our nonstandard implementation: Then it is his right to have the changes he needs.

b) Our misunderstanding: Then it is his right to have the changes he needs.

c) He found that it is better: We should evaluate the request and accept or reject.

4-Scheduling of future story in the current iteration: Sometimes customer shows his urgent need for a future (scheduled in one of the coming iterations) story to be done in the current iteration. Customer has to choose between one of the following:

a) Wait for the current iteration to finish and schedule the story in the next iteration plan

b) Replace the story with one or more equivalent (in size) stories.

5-Introducing new story or scenario: Customer may try to add new stories and business scenarios to the previously estimated and scheduled stories. Customer has to choose between one of the following:

a) Replace the story with one or more equivalent (in size) stories and drop the replaced stories from the release plan.

b) Open a story list for the next release plan (if not already opened) and add the new story to it.

6-Customer communication is too rapid and causes too much overhead: This needs careful handling by the team leader. Our suggestion is to fix two times per week for customer meeting. Always send next meeting agenda and needed documents, links,.. customer before meetings by an adequate time. This will give him time for preparing his requirements and limits his interruptions.

14-Testing

Testing (QC) process contains the following components:

1-Tester must attend every iteration planning meeting to take the detailed discussion on each story in the current iteration.

2-At the end of each iteration tester must (manually) test the developed stories during this iteration before deployment.

3-During the next iteration tester must write (or record) test scripts (*using selenium)* for the previously manually tested stories and store them in *Bromine.*

4-Tester cannot take more than 3 projects per two-weeks iteration. Each project will be assigned 3days to do manual and automated testing with one free day to handle exceptions.

15-Team work

1. Team must have at least **5** hours of intersection of their time at eSpace and leader has to dedicate not less than quarter of his day time and not more than half of his day time for process following up activities.
2. Team members should increase the communication to achieve the following:
    a) **Knowledge** and news sharing and updating.
    b) **Keep** their eyes on the **delivery time**. If there is a possibility for missing any deadline, the new schedule must be **communicated** early and clearly to all parties.
    c) **Solve** problems and take **decisions**.
    d) **Respond** to customer changes.

3. Tester checks the code unit tests coverage using the code coverage tool (*RCov*) and check the integrity of the system using integration tests at the end of each iteration.
4. Team is highly encouraged to focus on producing reusable components while working in their project. If They found some code that may be re-factored to be reusable they are encouraged to do this.
5. Any project must use eSpace templates, official components, libraries whenever exist and possible.
6. Team must write code in eSpace standard coding style. See “Enabling next effort” section below.
7. CTO and Tech directors have to assign a fixed time for each project to:
    a) Monitor and enhance the technical level of team members.
    b) Help in hard problems.
    c) Make sure that company goals are achieved.
    d) Encourage reusable components extraction and enriching our components library, code base, knowledge base and open source contributions.
    e) Recommend blog posts and sessions.
8. Project Measurement report has to be sent weekly, on Wednesdays, to services manager to follow up projects progresses.

16-Enabling next effort

Agile team main focus is to deliver the current project to the customer on time, on budget and with high quality. Team's second priority is to enable company next effort. The following activities will help in achieving this target:

1-Code review: CTO and Tech directors must have a periodical code review to detect the common problems in coding style and system design.

2-Code best practices and standard style: eSpace maintains and updates a list of important code practices that must be followed by every developer.

3-Component extraction: CTO and Tech directors should encourage teams to extract important components and build eSpace library.

4-Measurements: See measurements below.

5-Documentation: See documentation below.

8-Templates: eSpace maintains and updates templates to start projects quickly. All of the projects should use the existing templates whenever possible.

17-Measurements

The following measurements must be taken for each project (per iteration):

1-Team Velocity

2-Total time taken in project related activities categories (development, fixing bugs, meetings,...)

3-Current bug count (Severe, Major, Normal and Trivial)

4-Test coverage

18-Education

a) If the team has some valuable achievement they can publish this achievement via:
1.Blog posts.
2.Wiki entries.
3.Sessions.

b) Tech Talk is a mandatory session takes place every Thursday. It is a 1~2 hour session and may be one of the following:
1-Presentation
2-Exam
3-Code walk through
4-Ideas discussion

19-Project Documentation

Documentation is part of our projects quality. Developers have to deliver 4 documents for every project:

1- Vision of the system, Basic scenarios and Business logic.

2- Environment and setup: Describes server configurations, database, versions, system deployments, 3rd party packages and plugins and external systems.

3- Architecture design
1. Patterns, frameworks, models, important data-structures, ...
2. Main modules: used components, their interaction, main algorithms, assumption,...

Note: Skip architecture design If it's a pure normal MVC.

4- An auto-generated code documentation: by *rdoc.* Developers have to describe the code by comments.

5- Automated Unit tests and UI functional tests as dynamic documentation of the code.

6- Any other required documents by the customer must be handled as normal requirements and go through the normal work flow starting from estimation and ending with delivering. See “Iteration work flow” above.

20-Handover to support

Handover process is critical for smooth and successful support. The following are required from the development team that has just finished a project and it will be handovered to the support team:

a) Code coverage: not less than 80%

b) Documentation see “project documentation” above

c) Handover walk through.

d) 4 hours/week for 1 month after handover.

e) SVN contains the latest code. Code must be there in the form of (Check Out-Build-Run Tests-Go). This means you have to mock every external system and make it easy to switch between testing and production environments.

f) SVN follows eSpace standard structure.

g) Business handover: Contact persons, leftovers, ...

21-Project States

Any project may be in one of the following states:

1- **Development** (using *staging* server)

2- **Production only** (using *production* server)

3- **Production**/**support** (using *production* server and *amazon* server for testing)

4- **Production**/**development** (using *production* server and *staging* server for new features)

5- **Production**/**support**/**development** (using *production* server and *staging* server for new features and testing).

The only exception is the pure CMS projects. For these projects Production = Staging. Testing take place on developer's machines.