<Project name>

Software Development Plan

Document prepared by: Alexander Kozlinski
Last updated: 17 April 2004

Approved by:

[The project sponsor and key stakeholders should formally approve the Plan before the project team proceeds.]

Name, Title _____”___”_____

Name, Title _____”___”_____

Name, Title _____”___”_____

Name, Title _____”___”_____

Contact information: Alexander V. Kozlinski, mailto:alexkozlinski@uni-mod.com

Contents

Important note

This Software Development Plan is developed as a supplement to the chosen life cycle model and the Product Business Specification. The document contents are strictly limited to:

- Changes in the chosen life cycle model processes
- Facts specific to the project

1. Project definition

1.1. Scope

[Determine, which part of the product, defined in the Product Business Specifications, shall be developed within framework of this project.]

1.2. Timeframes

Start date:

End date:

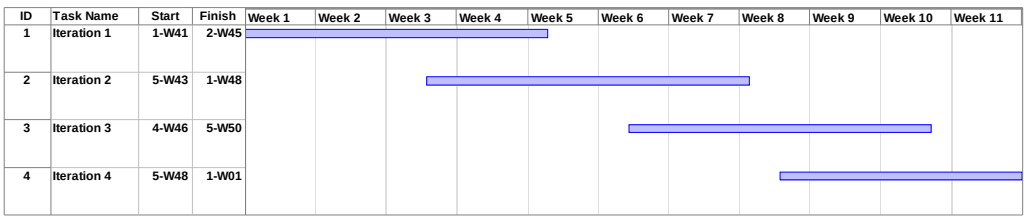
1.3. Iteration plan

[High—level iteration plan, showing the main milestones: iteration releases, iterations' scope. Iteration plan should be revised and agreed upon after each iteration or during iterations.

Use project initial data as starting point to develop project process structure. If initial data are well structured they should used as the 1st iteration of the project process structure.

Otherwise collect available data, number these data structure and use it to develop 1st draft of the project process structure.

1.3.1. Schedule



[For the everyday management, schedule could be maintained as a separate document.]

1.3.2. PBS—Iteration matrix

[List of the Product Business specifications to be realized on Iteration X.]

PBS#	PBS Description	Iteration 1	Iteration 1	Iteration 1	Iteration 1
		X			
			X		

[For the everyday management, matrix could be maintained as a separate document.]

2. Software life cycle model

[Reference the life cycle model (such as ISO/IEC 12207 or UMP) , or describe changes in the referenced model, or describe the model.]

2.1. Project process hierarchy changes

[Show the changes to the referenced model Process—Activity—Task hierarchy, or describe the hierarchy.]

2.2. Task description changes

[Describe the changes in the referenced model Tasks, or describe the Tasks.]

3. Methods and tools

[Method for each Process / Activity should be defined by applicable descriptions, such as:

- *Industry standards*
- *Published methodologies*
- *Company—wide standards and guides*
- *Project—specific guides, conventions and recommendations*
- *Modeling methodologies*
- *Programming languages*
- *etc.*

The rationale of choices could be added in this section.]

3.1. Development process

Process / Activity	Method	Model	Tool
Requirements collection	*[Reference to the method description if applicable]*	*[Reference to the model description if applicable]*	*[Reference to the tool developers guide]*
Analysis and design			
User interface design			
Programming			
Testing			

3.2. Support processes

Process / Activity	Method	Model	Tool
Documentation	*[Reference to the method description if applicable]*	*[Reference to the model description if applicable]*	*[Reference to the tool developers guide]*
Configuration management			
Content management			
Quality assurance			
Problem resolution			

3.3. Organizational processes

Process / Activity	Method	Model	Tool
Management	*[Reference to the method description if applicable]*	*[Reference to the model description if applicable]*	*[Reference to the tool developers guide]*
Infrastructure			
Improvement			

4. Support processes

4.1. Project Documentation

4.1.1. Naming conventions

[Define the document's header contents (e.g. title, subtitle, authors, project, subsystem, etc.), names' format and rules to compose the names.]

4.1.2. Mandatory sections

[List the mandatory sections, such as purpose, scope, document users, copyrights, etc. and define the requirements to the sections' content.]

4.1.3. Project documentation set

[Put here references to the developed documentation templates. Explain here any particular documents' features which are not clear from the document's templates itself, chosen life cycle model and other sections of this plan. See sample table below:]

Code	Document	Type	Reference to template
Guides			
	Software requirements analysis		
	UI prototyping		
	Software architecture design		
	Software subsystem design		
	Programming style		
	Software verification & evaluation		
	Software configuration management		
Project documentation			
	Software development plan		
	Product business specifications		
	User interface design		
	Software requirements specifications supplement		
	Software architecture design supplement		
	Software subsystem design supplement		
	Software Implementation model supplement		
	Software configuration management plan		
	Software change request specification		
	Software build description		
	Software qualification test plan		
	Software qualification protocol		
Checklists			
	Task implementation		
	SRS evaluation		
	UI prototype evaluation		
	SAD evaluation		
	SD evaluation		
	Code inspection		
	Subsystem evaluation		
	Software qualification test plan		
	Configuration management plan evaluation		
	Integration evaluation		

Code	Document	Type	Reference to template
	Qualification testing evaluation & verification		
	Documentation inspection		
	Change request evaluation		

4.1.4. Access rules

4.2. User Documentation

4.2.1. Naming conventions

4.2.2. Mandatory sections

4.2.3. Documentation set

Code	Document	Type	Reference to template

4.3. Change management

[Describe how the requested changes will be managed: collected, analyzed and implemented. Or reference to separate document.]

4.4. Validation plan

[Define the necessity of the validation.]

4.4.1. Validation goals

[Define validation goals for the Validators.]

4.4.2. Validation package

[Define the software packages which should be validated]

4.4.3. Validation procedures

4.4.3.1. Validators choosing criteria

[Describe here requirements to validators, hardware and software requirements, communications, etc.]

4.4.3.2. Validators motivating

4.4.3.3. Software package distribution

[Define how the package will be delivered to Validators]

4.4.3.4. Recommendation for validation

[Describe the directions on validation methods, tools, etc., for validators.]

4.4.3.5. Validation results collecting

[This is the part of the development task 5.3.4.UMP2: Collect product requirements. Define here how the results of validation should be collected.]

4.4.4. Validation timeframes

4.5. Problem resolution process

4.5.1. Problem classification

[Describe the most common problem types and how they could be identified. For examples, it could be quite general (any kind of problem) or specific technical and management types of problem. Each problem types should be addressed by resolution process.

4.5.2. Problem prioritization

[Describe problem prioritization rules.]

4.5.3. Problem initiation

[Define who, under which circumstances and how should initiate the problem.]

4.5.4. Problem analysis

[Describe who and what should analyze: e.g. problem reasons, impact etc.]

4.5.5. Problem discussion

[Define how the problem could be discussed: the roles of discussion participants, timeframes techniques, etc. For example, the joint review techniques could be used.]

4.5.6. Problem closure

[Describe the procedure of closing the problem.]

5. Organizational processes

5.1. Management process

5.1.1. Management plan

5.1.2. Team management

5.1.2.1. Developers' roles matrix

[Assign project process roles to developers.]

Developer name	Analyst	Architect	Configuration Manger	Database designer	Development manager	Designer	Graphics designer	Infrastructure administrator	Integrator	Product Manager	Programmer	Project manager	Tester	User interface designer

5.1.2.2. Organizational structure

[Show Organization chart for all individual project participants.]

5.1.3. Communication plan

[Define who, when, how and what should communicate and what kind of communication is off limits to whom.]

5.1.3.1. Communication levels

[Define who should communicate whom and which communication is restricted. For example, these communication levels could be assigned or even made mandatory:

- *communication between the project team and the project manager*
- *communication between the project manager and the rest of the participants*
- *communication between user representative and developer*

and the following communication could be limited or even disallowed:

- *communication from the developers to customers and sponsors*

Define the communication goals for each level. For example, it is extremely important to manage the project participants' expectations.]

5.1.3.2. Communication levels description

[Define the information, which should be communicated at each communication level. It could be as follows:

- *Project status information (as a rule, this is a must)*
- *Issues to be resolved*
- *Technical information*

Define the information status: mandatory, informational or event—triggered. Examples of mandatory information:

- *status reports*
- *budget reports, and*
- *legal and auditing requirements.*

Examples of informational communications:

- *project documentation*
- *frequently asked questions.*

Example of status report content:

- *whether the project is on track*
- *adherence to the project's budget and schedule*
- *progress since the last reporting period*
- *planned targets for the next period*
- *new risks*
- *current issues*
- *current scope change requests*

]

5.1.3.3. Communication channels

[Define the communication channels which should be used for each communication level and formats of the information communicated. Examples:

- *Project web site for publishing project documentation, deliverables, etc.*
- *Project discussion forum to communicate (and resolve) technical issues*
- *E-mail to communicate with consultants*
- *"Ready to print documents" to communicate with users*

]

5.1.4. Risk Management

Risk factor	Loss cost	Probability	Rank	Risk Minimizing Measures	Contingency Actions
			*[Kind of Loss*Probability measure]*		

5.1.5. Execution & Control

5.1.6. Project closure

5.2. Infrastructure process

See Infrastructure plan document.

5.3. Improvement process plan

5.3.1. Improvement events

5.3.2. Process assessment procedures

5.3.3. Process improvement procedures

UMP end—user license agreement (UMP—EULA)

1. Permission is hereby granted, free of charge, to any individual or entity obtaining a copy of this software and associated documentation files (the **Toolkit**), to deal in the **Toolkit** and to engage in any actions related to same without any restrictions whatsoever.

2. If you distribute, sublicense, publicly display, publicly and/or digitally perform, publish and/or sell copies of the **Toolkit** or any Derivative Works or Collective Works, you must include in all copies or substantial portions of the **Toolkit** the above copyright notice and this permission notice.

3. **The Toolkit is provided "as is", without warranty of any kind, express or implied, including, but not limited to, the warranties of merchantability, fitness for a particular purpose and non• infringement. In no event shall the authors and/or copyright holders be liable for any claim, damages or other liability, whether in an action of contract, tort or otherwise, arising from, out of, or in connection with, the Toolkit or the use or other dealings in the Toolkit.**

***** end of UMP—EULA *****

The UMP—EULA is based on the MIT License and Creative Commons License.