

The XML Interface of Project Observatory

The open pathway for project information

by

Gary M. Cole

CEO, Marin Research, Inc.

The XML Interface of Project Observatory

The Open Pathway for Project Information

Copyright (c) 2001 by Marin Research, Inc.

Revision A

Project Observatory™, Project Telemetry™, Project Gateway™, Project Directed Workflow™, ProjectGo!™, and the distinctive Gateway globe are trademarks of Marin Research, Inc. "Total Project Information. All the Time. Everywhere."™ and "Connecting Your Project Community"™ and "Bringing Projects into Focus"™ and "Observatory Pictures"™ are service marks of Marin Research, Inc. Lotus® and Lotus Notes® and Domino® are registered trademarks of Lotus Development Corporation. Windows® is a trademark, Internet Explorer® and Project® are registered trademarks of Microsoft. Netscape Navigator® is a trademark of Netscape Communications. PalmPilot® is a registered trademark of Palm Corporation. WorkPad® is a registered trademark of IBM Corporation.

Marin Research, Inc.

100 Larkspur Landing Circle, Suite 114

Larkspur, California 94939

info@marinres.com

<http://www.marinres.com>

I. Introduction

II. The Document Formats

III. Creating XML Documents

VI. The ProjectAbstract Document

V. The ProjectAmendment Document

Appendix A: The ProjectAbstract DTD

Appendix B: The ProjectAmendment DTD

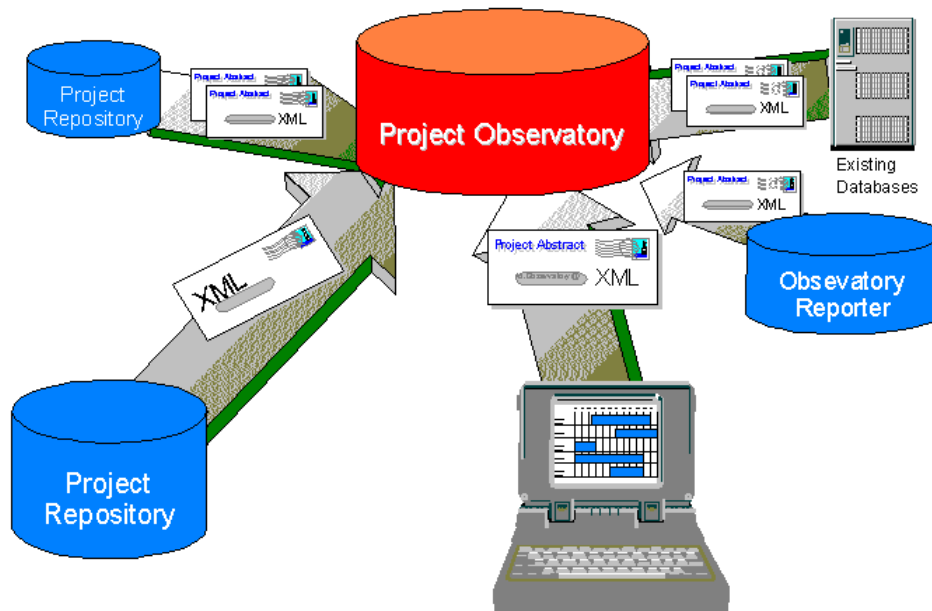
Also available from Marin Research:

The Architecture of Project Observatory

The Project Observatory System Guide

I. Introduction

Project Observatory uses a fundamentally simple process. People, assisted by software, report the status and resource requirements of projects by sending "reports" to the Observatory as XML formatted messages. Typically these are sent by email, but they can also be posted directly to a web page. The Observatory reads these messages and maintains a living database of current and historical project and resource information to support the needs of project, program and resource managers.



We have designed this system so that anyone can provide information to Observatory by simply following the XML data type definition that is documented herein. This means that you can get information from existing corporate databases, project scheduling systems and mobile reporting devices. You can also use these instructions to decode messages produced by Observatory compliant sources.

XML is the new standard for the communication of information between computer systems. XML itself is a framework of standard rules that allow application designers to package structured information into a text format. XML promotes the open interconnection of application systems. For more information see <http://www.w3c.org/XML>.

To connect new data sources to the Observatory system you need access to basic project facts (whether from a database or a person) and a programming tool that can generate text messages. Any programmable device, from pocket calculator to a supercomputer, can be used to create XML compliant data messages that can be processed by the Observatory system.

This document explains the semantics of the information that is transmitted in the XML Data Type Descriptions "ProjectAbstract" and "ProjectAmendment". It is intended for the use of analysts and programmers who are implementing custom data sources for the Project Observatory system. If you have questions on any of these variables, please contact Marin Research for more information (support@marinres.com).

II. The Document Formats

The messages that are understood by Project Observatory contain a series of tagged elements with predefined properties for representing project information. You can create text documents of these formats with any programming tool. The primary task is to determine what data do you have access to and how it should be mapped to the specific attributes defined in the Data Type Definition.

The XML Data Type Descriptions defines the structure for formatting text documents. The formatting consists of a series of tags in the form <tag attributes >content</tag> or <tag attributes />. Each tag has a matching end /> or </tag> somewhere later in the file. The full text of these definitions is found in the appendix.

Project Observatory understands two DTD formats. One of these is called a ProjectAbstract and the other a ProjectAmendment.

The ProjectAbstract Format

This is the format of a full project report. Not all projects will contain all of the possible kinds of information.

There is an overall tag set called a **projectabstract** to transmit the identification of the project.

Within this structure are some or all of the following tagged sections.

- **projectstatistics** to transmit the top level facts such as workhours and cost.
- **projectdescription** to transmit the written description of the project.
- **projectreport** to transmit the written status of the project
- **projectcommitment** to transmit the name and amount of resources used over time.
- **projectissue** to transmit the status of "issues" e.g. problems that management needs to know about.
- **projectevent** to transmit the status of project milestones.
- **projectdocument** to transmit the title and location of supporting documents about the project.

The ProjectAmendment Format

This DTD is even simpler than the projectabstract. It is use to "amend" project reports with additional facts that might not be directly available to the sender of the projectabstract.

There is an overall tag set called a **projectamendment** to transmit the identification of the project.

- **projectstatistics** to transmit the amended facts such as actual cost.

Transferring Information to the Project Observatory Database

1. Once the project abstract has been prepared as a local file, simply attach it to an email message and route that message to the Observatory Database's public email address. The subject of the message should contain the project name for clarity.
2. Use the "Submit Report" option on the Observatory Management Navigator. It will provide a form where you can attach the XML file for submission.

III. Creating XML Documents

Handling Dates

In general datetimes are communicated as GMT (Universal Time). This allows the proper integration of reports that come from different timezones.

To convert US Standard Time Zones to GMT subtract 8 from Pacific, 7 from Western, 6 from Central, 5 from Eastern. Subtract one less hour during daylight savings time.

If the attribute zone="N" is specified, then the datetimes will be assumed to be in the same time zone as the server and will not be converted. This is used, for example, by the Observatory Reporter for MS Project because Project does not have a timezone concept. In these reports all dates and times are taken literally.

Dates and times are rendered into text strings as follows.

`datetime="yyyymmdd:hhmmss"`

Important - leading zeros must be used in all subfields. Hours are entered in 24 hour format. (14 is 2 pm)

e.g. `datetime="20010205:142546"` - Feb 5, 2001 at 2:25.46 PM GMT (Feb 5, at 6:25 PST)

Where allowed, dates without times are rendered as:

`datetime="yyyymmdd"`

e.g. the `date="20010205"` - Feb 5, 2001, no time specified.

Handling Numeric Values

Number attributes should be written as integers or decimal fractions. Negative values should be prefixed with a "-" minus sign. Exponent formats are not supported. If decimal fractions are needed, then the decimal point or comma may be used as a decimal separator no matter what the local setting of the system. Some programming libraries use the OS convention and some do not. The use of a decimal point is preferred.

Handling Text and Special Characters

Because XML is a tagged markup language, certain characters cannot be used with attribute values because they would be misinterpreted. Therefore the following entity substitutions must be used within attribute text strings. In no case should any text be used as an attribute that has not been scanned and converted using these substitutions.

`<` for "<"

`>` for ">"

`&` for "&"

`'` for "'"

`"` for "\""

`&cr;` for character 13

`&lf;` for character 10

Example:

original text: "Bill's a great guy" Sharon said.

converted text: `"Bill's a great guy"; Sharon said.`

Note: All attribute and body character strings must be processed with a routine to replace certain special characters with entity representations.

IV. The Project Abstract Document

A Comprehensive Report

The system assumes that each projectabstract is a complete report about that project. This avoids the need to perform complicated "deletion" rituals to remove outdated information.

Therefore you must specify all of the information to be used every time the project is resent. Examples are Project Name, Program Name, Sponsor name, projectApproval, Description, Project Documents etc. To do this you will need to maintain some persistent state information at the source of the reports. This can be done using User defined variable (available in MS project for example) or INI files or additional database entries. As an absolute minimum, the project ID attribute must be preserved and resent with every report so that it is correctly used.

Structure of a Project Abstract

<XML DTD header>

<projectabstract

<projectstatistics />

<projectdescription>body</projectdescription>

repeat for each issue...

<projectissue>body</projectissue>

repeat for each project document...

<projectdocument />

repeat for each resource...

<projectcommitment name="resourcename">

repeat for each week

<period

</projectcommitment>

</projectabstract>

A minimal project abstract must have a projectabstract element and should have a projectstatistics element. (The xml header has been omitted)

```
<projectabstract id="123456" name="My Project" sourceRepositoryTitle="My Database" reporttime="20010205:140000" >
```

```
<projectstatistics projectStartDate="20010101:160000"
projectFinishDate="20010301:160000" asgTotalWork="100"
asgActualWork="0" asgWorkRem="100" >
```

```
</projectabstract>
```

ProjectAbstract Attributes

The attributes on the Project Abstract Element provide the critical structure for processing the report and integrating it into the Observatory database. The fields which are marked with an "*" in the list below are very important and should always be provided.

```
<projectabstract id="123456" name="My Project" sourceRepositoryTitle="My Database"
reporttime="20010205:140000" >
```

Attribute Name	Kind	Example
id **	Text string value of a unique integer number up to 14 digits. No two projects should have the same number.	id="1234567"
name **	project name (text)	name="My Project"
reportTime	text date	reportTime="20010101:140000"
senderType	text code value	senderType="PGL"
senderVersion	text	senderVersion="A"
sourceRepositoryTitle **	text	sourceRepositoryTitle="Western Projects"
sourceRepositoryFile	text	sourceRepositoryFile="WP.NSF"
sourceRepositoryServer	text	sourceRepositoryServer="MyServer"
programName	text, multiple levels separated by backslash	programName="Research Lab\Development Work"
managerSendTo **	text, user name (if manager is in the Notes name and address book) or full email address	managerSendTo="cjones@foonet.net" or managerSendTo="Charles Jones"
sponsor	text	sponsor="Marketing"
projectApproval **	text value of an integer number between 1 and 6	projectApproval="2"
approvedUntil	text date	approvedUntil="20020101:000000"
senderEnvironment	text	not currently used
zone	text	omitted or zone=N" (see instructions)

Table 1: Attributes for the ProjectAbstract Element. ** indicates required attribute.

id **

This is a string form of a unique integer number representing the project. It is normally formed by expressing the creation date of the project as an integer number of seconds since 1980. Project Gateway, Observatory Reporter, and the MSP Observatory Link all use this method for assigning IDs. As a result, numbers below 500,000,000 will never be found from these sources. The important thing to ensure is that this number is retained at the source so that all reports about this project use the same ID. This is the only unique identifier for the project.

name **

This is the text name of the project. While not a required field, you should always provide a name.

If you do not provide a project name, the Observatory message processor will create one in the form "Project:uniqueprojectid" e.g. Project:234413213

reportTime

This is the date and time of this report about the project or the time for which the data being transmitted is valid (the data date). This is usually the time that the source information was last modified. If not provided the Observatory history and graphing mechanisms will not operate correctly. This value should also be supplied as a member of the projectstatistics element.

senderType

This is a string used to indicate the client code this is generating the message. The string value "PGL" and "OR" are predefined for Project Gateway and Observatory Reporter and should not be used.

senderVersion

This is a version identifier for the senderType.

sourceRepositoryTitle **

This is a string name for the source (origin) of the report. Each source must have a unique name. The source name must be provided with each subsequent report about the project. For example, if you are exporting reports from a regional database of projects, you might name the source as "WesternProjects". The name must be unique among all sources feeding the observatory. The name will show as an Origin in the Origin Worksheet. It must not contain any "\" characters.

Note: The XML Repository Link agent will set this from the Repository Database Title unless an option is set in the agent declaration to use a different name. The Observatory Reporter application has its sourceRepositoryTitle set by the system administrator in the Administration document. The Microsoft Project Link macro gets its sourceRepositoryName from a required field on the Observatory Link dialog box.

sourceRepositoryFile

This will be the name of the source data file. This should be the complete file name from the root of the server's local data directory. This information is recorded in project reports but it is not required for any of the system functions.

sourceRepositoryServer

This will be the name of the server on which the source data is maintained. This should be the full server name. This information is recorded in project reports but it is not required for any of the system functions.

programName

This is a string identifying the program association used in the source database. If there is only one project coming from this source, or if there is no logical grouping of projects, then this should be omitted. The format of this string should contain "\" characters if multiple levels are required. Each level of program names will appear in the Program Assignment Worksheet as a subcategory for projects under the source name. Once the program is specified, it should be repeated in all subsequent messages for this project.

managerSendTo

This should be the user name of the project manager (if that person is in the name and address book used by the observatory server) or the person's email address if it is not. This name is used to send email back to the project manager and to provide access to the observatory project reports and forms.

The Observatory has a modal option on each project that determines if the SendTo field can be reset by the contents of this parameter.

sponsor CDATA #IMPLIED

The sponsor is a string identifying the sponsor of the project. This is optional.

projectApproval

The projectApproval is a string which must be of the form of a single digit number with the following meaning. If not specified, then "1" is set by default.

<code>projectApproval="1"</code>	Approved
<code>projectApproval="2"</code>	Approved until the date given in approvedUntil attribute
<code>projectApproval="3"</code>	Proposed
<code>projectApproval="4"</code>	Cancelled
<code>projectApproval="5"</code>	Finished
<code>projectApproval="6"</code>	On Hold after the date given in the approvedUntil attribute

Note: How Observatory handles the projectApproval attribute

Commitments to a proposed project, or which fall after the approvedUntil date are recorded in the "work proposed" array and are imaged in blue. Regular commitments are imaged in yellow.

Cancelled Projects are automatically archived according to rules set by the Observatory Administrator.

Finished Projects are automatically archived according to rules set by the Observatory Administrator.

When a project is to be deleted from a source system, the reporting agent should send a final report indicating the final state (Finished or Cancelled). Otherwise the Observatory will send reminders about missing reports to the project and program managers until it declares the project as abandoned (according to rules set by the Observatory Administrator) and archives it.

approvedUntil

If the projectApproval has the values of 2 or 6 then this must be a string date representation of the associated date. The interpretation of this date is to create a dividing line between committed and proposed work.

senderEnvironment

This is not currently used and may be omitted.

zone

This parameter may be set to the text value "N" to signal that all datetimes are in the same zone as that of the server. Otherwise all datetimes are assumed to be in GMT.

ProjectStatistics Attributes

The project statistics element contains the primary measures of the project. While all of these are optional in a strict XML sense, the Observatory expects at least a minimal set of information for real projects. These minimal attributes are marked with "***" in the list below.

Attribute	Kind	Example
asgTotalWork**	number of hours	asgTotalWork="100"
asgCount	integer	asgCount="20"
asgStarted	integer	asgStarted="10"
asgFinished	integer	asgFinished="7"
asgOverdue	integer	asgOverdue="1"
asgActualWork **	number of hours	asgActualWork="25.2"
asgWorkRem **	number of hours	asgWorkRem="74.8"
asgOverdueWork	number of hours	asgOverdueWork="5"
asgRefWork	number of hours	asgRefWork="80"
asgEarnedWork	number of hours	asgEarnedWork="25"
keyEventCount	integer	keyEventCount="5"
keyEventFinishedCount	integer	keyEventFinishedCount="1"
asgLatestFinish**	date	asgLatestFinish="20010401:140000"
targetCost CDATA	cost number (in source currency)	targetCost="4500"
targetFinishDate	date	targetFinishDate="20010412:140000"
targetEffort	number of hours	targetEffort="150"
issuesOverdueCount	integer	issuesOverdueCount="1"
issuesOpenCount	integer	issuesOpenCount="6"
reportTime **	date for which information is valid.	reportTime="20010207:200700"
projectStartDate**	date	projectStartDate="20001205:140000"

projectFinishDate **	date	projectFinishDate="20010401:140000"
refStartDate	date	refStartDate="20001205:140000"
refFinishDate	date	refFinishDate="20010301:140000"
refWork	number of hours	refWork="80"
refCost	cost number (in source currency)	refCost="3500"
outputCount	integer	outputCount="3"
outputFinishedCount	integer	outputFinishedCount="0"
actualCost **	cost number (in source currency)	actualCost="1500"
totalCost **	cost number (in source currency)	totalCost="5000"
User defined statistics	text, number or date	usrPriorityCode="3A"

Table 2: Attributes of the Project Statistics Element. ** indicates that this is required.

asgTotalWork **

This is the total work on the project estimated at completion expressed in decimal hours.

asgCount

This is the number of assignments in the project expressed as an integer. If the project plan is not defined in terms of tasks and assignments, then omit this parameter. If only tasks are used, count tasks, but if assignments are defined, count only assignments and count 1 for each task without any assignments.

asgStarted

This is the number of assignments that are started. This must be no larger than asgCount

asgFinished

This is the number of assignments that are finished. This must be no larger than asgCount and no smaller than asgStarted

asgOverdue

This is the number of assignments that are not started but should have been as of the date of the report. It must be no larger than asgCount

asgActualWork **

This is the number of hours of actualWork completed as of the report.

asgWorkRem

This is the number of hours of work remaining to be done by the completion of the project.

asgOverdueWork

This is the number of hours of work which should have been done by the date of the report, but have not been done.

asgRefWork

This is the number of hours of work for the project in the baseline plan of that project.

asgEarnedWork

This is the number of hours of work for the project in the baseline plan of the project associated with work that is completed.

keyEventCount

This is the number of milestones or assignments marked as key events in the project.

keyEventFinishedCount

This is the number of milestones or assignments marked as key events that are done.

asgLatestFinish **

This is the latest completion date of the latest assignment in the project (project finish)

targetCost

This is the management goal for cost at completion.

targetFinishDate

This is the management goal for project completion date.

targetEffort

This is the management goal for project effort hours at completion.

issuesOverdueCount

This is the number of issues that are currently overdue for resolution.

issuesOpenCount

This is the number of issues open on the project.

reportTime **

This is the datetime for which the reported data is supplied. Normally it is the time the data is entered.

projectStartDate

This is the actual or planned start date for the project.

projectFinishDate

This is the actual or planned finish date for the project. It is required if asgLatestFinish is not provided.

refStartDate

This is the start date of the project in its baseline plan.

refFinishDate

This is the finish date of the project in its baseline plan.

refWork

This is the total effort at completion of the project in its baseline plan.

refCost

This is the total cost of completion of the project in its baseline plan in the designated currency specified for that origin.

outputCount

This is the number of outputs (project deliverables) expected at the completion of the project.

outputFinishedCount

This is the number of outputs (project deliverables) that are actually complete as of the date of the report.

actualCost **

This is the actual cost of the project as of the date of the report expressed in the designated currency specified for that origin.

totalCost **

This is the expected cost at completion of the project as of the date of the report in the designated currency specified for that origin.

User defined statistics

The statistics section of the abstract can be extended to support additional named fields. Each such field must be associated with a Custom XML Field Form in the Observatory database.

Attributes which are not recognized are ignored. As a result, it is very important that all attribute names be spelled correctly.

Naming conventions for user defined statistics attributes.

We have adopted the convention that all user defined statistics will be named with the prefix `usr`. The CustomXML field forms provided by Observatory, Gateway, and Reporter all use this convention.

Note that the way in which the attribute is interpreted and recorded is defined by the CustomXML field form which you define in the Observatory. There is a limit to the number of trended fields(8) and to the total number of user defined fields (50).

Example: `usrSpecialPriority`**Understanding the use of Target and Reference attributes.**

The ref Attributes (`refStart`, `refFinish`, `refWork`, `refCost`) are values which should be derived from the current baseline plan.

The target Attributes (`targetCost`, `targetDate`, `targetEffort`) are goal values specified for the project by the management or customer. They may or may not be consistent with the current or baseline schedule. Target attribute values can be specified or overridden by Observatory users.

NOTE: There is no value in making up numbers. Do not send target or reference values unless they are meaningful information.

The Project Event Element

A project event element should be created for each milestone in the project. Not all project systems will have a way of identifying milestones.

Format: `<projectevent attributes....>Name of Event</projectevent>`

Attribute	Kind	Example
index**	integer, assigned sequentially starting from 1 for each milestone within each project.	index="1"
plannedFinishDate**	date	plannedFinishDate="20010209:140000"
plannedStartDate	date (must be earlier or equal to planned finish date)	plannedStartDate="20010207:140000"
refFinishDate	date (finish date in baseline plan)	
actualStartDate	date (if event is started)	
actualFinishDate	date (if event is finished)	
percentComplete**	integer, 0 to 100	percentComplete="0"

Table 3: Attributes for projectevent element. ** indicates required attribute.

index

This is a text representation of an integer which should be assigned sequentially. The numbers are used only in the context of this report and do not need to be remembered for subsequent reports.

plannedFinishDate

The expected finish date of the event or milestone.

plannedStartDate

The expected start date of the event or milestone. May be the same as the finish date.

refFinishDate

The finish date of the milestone in the baseline plan if there is one specified.

actualStartDate

The actual start date of the event if it has been started.

actualFinishDate

The actual finish date of the event if it is actually finished.

percentComplete

The percent complete of the event. Must be 0 if the event is not started and must be 100 if the event is actually finished.

BODY

The body is the name of the milestone or event. It is the text that appears between the <projectevent ...> and </projectevent> tags.

In this example the milestone called "Phase 1 Complete" was reported as actually finished on 12/23/1999 at 1AM GMT.

```
<projectevent index="1" plannedFinishDate="19990930:000000"
plannedStartDate="19990929:160000" refFinishDate="TEST"
```

```
actualStartDate="19991222:170000" actualFinishDate="19991223:010000"
percentComplete="100">Phase 1 Complete"</projectevent>
```

ProjectIssue

A project issue is a problem that may require management attention. The issue is presented with a title and two lines of source generated text and a body of text which should be limited to no more than a paragraph for screen layout reasons. Not all project systems will have a way of providing a list of issues.

Attribute	Kind	Example
index**	integer, assigned sequentially starting from 1 for each issue within each project.	index="1"
dueDate**	date	dueDate="20010209:140000"
firstLine	text	firstLine=""
secondLine	text	secondLine=""
url	text in URL format	url="http://www.marinres.com"

Table 4: Attributes for projectissue element. ** indicates required attribute.

A project issue contains a header with up to 4 attributes and a body of text.

```
<projectissue index=1 dueDate="20010207:230000 firstLine="formatted text
A" secondLine="formatted text B" url="urltext">
```

body of issue

```
</projectissue>
```

index

The index is a positive number specified as a string. Index values should be sequential integers within a project beginning at 1.

dueDate

The dueDate is a datetime specified for the required resolution of the issue. If the currenttime is later than the given time, then the issue is considered to be overdue.

firstLine

This is a text string that is intended to be preformatted as follows. Note that the observatory system expects a ":" delimiter between the tag and the title. The contents of this line will be parsed and manipulated with the assumption that this is the format.

```
firstLine="Overdue Issue:Development of compatibility module is delayed"
```

secondLine

This is a text string that is intended to be preformatted as follows:

The observatory does not manipulate this text, so it does not need to be formatted this way. However the text will be displayed in the project report, so it should be no longer than a typical line and attractive in presentation.

```
secondLine="Importance: 3 Assigned to: mr x"
```

url

The url attribute is used by the Observatory reports to provide hyperlinks directed to the source documents defining the issue if those are available as web pages. This urls will be used literally and will not be parsed or otherwise transformed except to embed them in Anchor expressions. Remember that these attributes must be text converted as described.

```
url="http://207.20.108.66/PG5B1Demopg.NSF/$DefaultView/5AFD2E6175621FB9882569A60066FC25?OpenDocument"
```

Example:

```
<projectissue
index="1"
dueDate="20001206"
firstLine="Overdue Issue:Development of compatibility module is
delayed"
secondLine="Importance: 3 Assigned to: mr x"
url="http://207.20.108.66/PG5B1Demopg.NSF/$DefaultView/5AFD2E6175621FB98
82569A60066FC25?OpenDocument "
>
This is the text content for the issue.
</projectissue>
```

ProjectReport

This is used to transmit the written words of the project manager and a status phrase to describe the schedule situation. Note that the date attribute is used to identify the date of the written words which might be some time earlier than the statistics.

Attribute	Kind	Example
author **	text	author="Bill Jones"
date **	date	date="20010209:140000"
summary	text	summary="Slipped but Stable"

Table 5: Attributes for projectreport element. ** indicates required attribute.

author

This is the name of the author of the status report (generally the project manager)

date

This is the datetime of the status report information. It is the point in time that this written report is assumed to be accurate. It may not be the same as the reportTime of the Project Abstract.

summary

This is a short summary of the project status. We generally use one of a set of standard phrases: On Schedule, Behind Schedule, Ahead of Schedule, Slipping, Slipping but Stable.

The body of this element is the written report itself which will be copied in to the project report form unchanged except for special character processing.

Project Description

There are no attributes on this element. It is simply used to tag a description of the project.

```
<projectdescription>
```

```
This is the purpose of my project.....
```

```
.....
```

```
....
```

```
</projectdescription>
```

Project Document

This element is used to identify a specific document that is part of a project (such as a work plan, customer letter, etc.) The observatory will make a list of these documents and display the url provided so that observatory users can readily access the project materials. The url can point anywhere. These elements are generated by Project Gateway to make all "project documents" available in the Observatory. They are optional.

Attribute	Kind	Example
author **	text	author="Bill Jones"
title **	date	date="20010209:140000"
status	text value	status="Y"
url	text	url="http://www.marinres.com/w hatnew.html"

Table 6: Attributes for projectdocument element. ** indicates required attribute.

url

This is the pointer to the document.

title

This is the document title or short description.

author

This is the name of the author.

status

This is the status of the document as a one character string. (N=Draft, Y=Final). The status field accepted but is not used in Project Observatory Version 2.0 but may be supported in later releases.

Example:

```
<projectdocument  
url="http://207.20.108.66/PG5B1Demopg.NSF/$DefaultView/36260957EBB413138  
82569180077AB3A?OpenDocument"  
title="System Plan"  
author="Gary M. Cole/MarinResearch"  
status="N" />
```

Project Commitment

This is way you transfer resource requirements and actual work.

Each commitment contains a commitment element followed by one or more Period Elements

The period elements are used to report "how much" for each period. Generally a period is a week.

Example:

The resource named "Bill" works 10 hours in the first week and 2 hours in the second week.

```
<projectcommitment name="Bill">
<period sd="20010101" days="7" wa="600" wp="600">
<period sd="20010108" days="7" wp="120">
</projectcommitment>
```

Project Commitment Attributes

Attribute	Kind	Example
name**	text	name="Bill Jones"

Table 7: Attributes for projectcommitment element. ** indicates required attribute.

name

The name of the resource is specified. This should include the hierarchical group name if there is one. All resources in the project must have unique names. The name will be mapped by the observatory administrator into an observatory member or organization.

Period Attributes

A period tag is used to convey the effort and assignment transition counts of a resource for a timeperiod. Several period records are usually required to describe one commitment. For a one year project, you would typically have about one per week (per resource). So these are always the most common element in a Project Abstract.

Attribute	Kind	Example
sd **	Date for start of period.	sd="20010205"
days	Integer number of days that must be 7 or a multiple of 7. If not provided, 7 is assumed.	days="7"
wa	Text value of integer number of person-minutes of actual work in this period	wa="60" This is one person-hour (60 person-minutes) for the week of actual work
wp	Text value of integer number of person-minutes of work in the period (actual plus remaining)	wp="2400" This is 40 person-hours (2400 person-minutes) of scheduled

		work for this week. Includes the wa value.
wb	Text value of integer number of person-minutes of baseline plan work in this period	wb="1800" Report 30 person- hours of baseline work for the week.
sa	Text value of integer number of tasks/assignments actually started in this period	sa="2" Report that two assignments were started this week.
sp	Text value of integer number of tasks/assignments planned to start in this period (including those actually started)	sp="3" Report that three assignments were planned to start this week (includes the 2 actually started)
sb	Text value of integer number of tasks/assignments planned to start in this period in the baseline plan.	sb="5" Report that 5 assignments were supposed to start this week in the baseline plan.
fa	Text value of integer number of tasks/assignments actually finished in this period	fa="1" Report that one assignment was finished in this period.
fp	Text value of integer number of tasks/assignments planned to be finished in this period (including actuals)	fp="2" Report that two assignments were planned to be finished in this period including the one that was actually finished
fb	Text value of integer number of tasks/assignments planned to be finished in this period in the baseline plan	fb="5" Report that 5 assignments were supposed to be finished this period according to the baseline plan.

Table 8: Attributes for period element. ** indicates required attribute.

sd

This is the starting datetime of the period.

Project Observatory will take the given time and map it to the observatory period in which it falls. Observatory periods are always Monday-Sunday weeks, so it is best if the data is reported for the same dates.

days

This is the number of days in the period. If not specified, then 7 is assumed. The value must be a multiple of 7. If it is greater than 7, then the values will be assumed to apply to each week.

wa

Actual work expressed in integer work-minutes. A full 40 hour week spent on a project would be wa="2400"

wp

Planned work expressed in integer work-minutes. A half time 20 hour week planned for this project would be wp="1200" (20 hours x 60 minutes)

wb

Baseline (Reference) work expressed in integer work-minutes.

The following six parameters may be used if you have a task level schedule for this resource in this project. if you do not have tasks, then these values should not be specified. They are optional in any case.

sa

Number of tasks for this resource on this project actually started in this period.

sp

Number of tasks for this resource on this project planned to start in the period in the current project schedule.

sb

Number of tasks for this resource on this project planned to start in this period in the baseline schedule.

fa

Number of tasks for this resource on this project actually finished in this period.

fp

Number of tasks for this resource on this project planned to finish in the period in the current project schedule.

fb

Number of tasks for this resource on this project planned to start in the period in the baseline project schedule.

Computing the Period Information from Project Task Lists

Method 1: If your PM system provides the feature, have it compute a weekly histogram for each resource and then generate the project commitment element from this data. This method is used in the MSP Observatory Link macro.

Method 2: If you have a list of tasks for a resource you can compute the period work plan in an array and then generate the project commitment element from this array. This is a simplified version of the method is used in the Project Gateway XML Repository Link agent.

Step 1: Define an array that has one element for each week of the project timeframe. The index of the array will be a week number derived relative to the start of the project. Weeks should begin on Monday.

A typical resource assignment in a project management system will have the following information: Startdate, Finishdate, Totaleffort.

Step 2: For each assignment of a resource do the following

- a. Compute TWD=Find number of workdays between start and finish

- b. Compute $HPD = \text{Divide totalwork by TWD average workhours per day}$
- c. Compute $\text{StartWeekWD} = \text{number workdays in the week of the startday starting at the startday}$
- d. Compute $(\text{StartWeekWD} * HPD)$ and add to the array[startweek]
- e. For all intermediate weeks Compute $(HPD * 5)$ and add to the array[week]
- f. Compute $\text{EndWeekWD} = \text{number of workdays in the week of the finish day through the finish date.}$
- g. Compute $(\text{EndWeekWD} * HPD)$ and add to the array[endweek]

Step 3:

Write the `<periodcommitment name="Bill">` record

For all nonzero members of the array:

Write the period record `<period sd="(StartDayOfWeek)" days="7" wp=array[week]>`

Write the `</projectcommitment>` closing record

Remember that all the work values must be expressed in man-minutes (not hours). So multiple hours by 60 and round so that the total work is represented to the nearest minute. This means keeping track of the total minutes emitted and possibly adding or subtracting a minute from the final period to get the right total.

Skill

The skill element is an allowed component of the projectcommitment element. Its use, however, is discouraged because the Observatory provides an automated interview facility to collect similar information. The names used for skills should correspond to predefined Custom Skills. If they do not, then they should match one of the external skill names that has been predefined in the Custom Skills form.

You have the choice of providing the total of the person-minutes of work for each skill or declaring the skill usage to be a percentage of the total usage reported in the period records. There is no provision for defining the skill utilization on a time phased basis (i.e. skill utilization by period.)

Skill Attributes

Attribute	Kind	Example
wa	integer number of total person-minutes of actual work in this skill	wa="60"
wp	integer number of total person-minutes of work (actual plus remaining) in this skill	wp="2400"
wb	integer number of total person-minutes of baseline plan work in this skill.	wb="1800"
wq	integer number of total person-minutes of proposed work in this skill.	wq="90"
alloc	integer number 1 to 100	alloc="100" This is the percent of allocation of the resource on this skill. This is used in lieu of the explicit values of the w-attributes.
name**	text name of the skill. Must be one of predefined values.	name="Programming\Notes"

Table : Attributes for skill element. ** indicates required attribute.

V. The ProjectAmendment Document

A `<projectamendment>` is used to add or correct statistical information provided in a `projectabstract`. This was intended for the situation where specific project facts such as cost to date, were available from a corporate database. The `projectabstract` on the other hand, was created manually or from Microsoft project or some other tool which could not easily look up the cost data.

An amendment is a standing correction to a project report. First the project report is processed, then any changes that have been received via `projectamendments` are made to it. Once an amendment has been received, the facts that it asserts will be asserted forever until those same facts are changed by a later amendment.

IMPORTANT: The amendment has priority over the `projectabstract`. Your amendment should transfer **ONLY** the facts to be added or amended and not any others. In other words, if you want to update the actual cost, the `projectstatistics` part of the amendment should contain only one attribute i.e. `asgActualCost`. If you sent all the possible attributes in an amendment then they will replace all the statistics from the project report.

Any number of amendments can be sent from separate sources to amend the project report. Since you have no way of guaranteeing the sequence in which they are processed, you should ensure that no two sources transmit the same attribute. So long as each sender keeps to its own subject matter, the results will be predictable and useful.

You should not send amendments to projects that do not exist.

Structure of a `projectamendment`

xml header

```
<projectamendment .....>
<projectstatistics ..... />
</projectamendment>
```

Example: An existing project with the id 9299930 is in the observatory. The following will set the actualcost of that project to 23,002.01 in the current project currency units.

```
<projectamendment id="9299930" >
<projectstatistics asgActualCost="23002.01" />
</projectamendment>
```

Projectamendment Attributes

The allowable attributes are the same as those for a project abstract. Please note, however, that the "id" is the only parameter that is actually used by the Observatory software. The other parameters are recorded into the `ProjectAmendment` document, but are not put into the project report. Therefore, it is not possible to change the `projectname`, `programname`, `source`, etc. using an amendment. We suggest using the "id" and "name" attributes only.

Projectstatistics Attributes

The allowable attributes are the same as for the `projectstatistics` element of a `projectabstract` and are described in the preceding sections. Any statistic that is provided will override the same statistic provided in `projectabstracts`. Therefore you should assert only those which you are explicitly trying to transmit.

Never set "dummy values" in an amendment as this will prevent the real values from being reported in an projectabstract.

Appendix A. Project Abstract Document Type Definition

```
<?xml version="1.0"?>
<!DOCTYPE PROJECTABSTRACT [
<!-- document type definition copyright (c) 1998 by Marin Research -->
<!ENTITY lt "&#38;#60;">
<!ENTITY gt "&#62;">
<!ENTITY amp "&#38;#38;">
<!ENTITY apos "&#39;">
<!ENTITY quot "&#34;">
<!ENTITY lf "&#10;">
<!ENTITY cr "&#13;">
<!ELEMENT PROJECTABSTRACT
(projectstatistics*,projectdescription*,projectcommitment*,projectevent*
,projectissue*,projectreport*,projectdocument*)*>
<!ATTLIST PROJECTABSTRACT
  id CDATA #REQUIRED
  name CDATA #IMPLIED
  reportTime CDATA #IMPLIED
  sourceRepositoryTitle CDATA #IMPLIED
  sourceRepositoryFile CDATA #IMPLIED
  sourceRepositoryServer CDATA #IMPLIED
  programName CDATA #IMPLIED
  managerSendTo CDATA #IMPLIED
  senderType CDATA #IMPLIED
  senderVersion CDATA #IMPLIED
  sponsor CDATA #IMPLIED
  projectApproval CDATA #IMPLIED
  approvedUntil CDATA #IMPLIED
  senderEnvironment CDATA #IMPLIED
  zone CDATA #IMPLIED
>
<!ELEMENT projectcommitment (period*,skill*)*>
<!ATTLIST projectcommitment
  name CDATA #REQUIRED>
<!ELEMENT period EMPTY>
<!ATTLIST period
  sd CDATA #IMPLIED
  days CDATA #IMPLIED
  wa CDATA #IMPLIED
  wp CDATA #IMPLIED
  wb CDATA #IMPLIED
  sa CDATA #IMPLIED
  sp CDATA #IMPLIED
  sb CDATA #IMPLIED
  fa CDATA #IMPLIED
  fp CDATA #IMPLIED
  fb CDATA #IMPLIED
>
<!ELEMENT projectevent (#PCDATA)>
<!ATTLIST projectevent
  index CDATA #IMPLIED
  plannedFinishDate CDATA #IMPLIED
  plannedStartDate CDATA #IMPLIED
  refFinishDate CDATA #IMPLIED
  actualStartDate CDATA #IMPLIED
  actualFinishDate CDATA #IMPLIED
```

```
    percentComplete CDATA #IMPLIED
  >
  <!--ELEMENT projectissue (#PCDATA)-->
  <!--ATTLIST projectissue
    index CDATA #IMPLIED
    dueDate CDATA #IMPLIED
    firstLine CDATA #IMPLIED
    secondLine CDATA #IMPLIED
  >
  <!--ELEMENT projectreport (#PCDATA)-->
  <!--ATTLIST projectreport
    author CDATA #IMPLIED
    date CDATA #IMPLIED
    summary CDATA #IMPLIED
  >
  <!--ELEMENT projectdescription (#PCDATA)-->
  <!--ELEMENT projectdocument (#PCDATA)-->
  <!--ATTLIST projectdocument
    url CDATA #IMPLIED
    title CDATA #IMPLIED
    author CDATA #IMPLIED
    status CDATA #IMPLIED
  >
  <!--ELEMENT skill EMPTY-->
  <!--ATTLIST skill
    wa CDATA #IMPLIED
    wb CDATA #IMPLIED
    wp CDATA #IMPLIED
    wq CDATA #IMPLIED
    alloc CDATA #IMPLIED
    name CDATA #REQUIRED
  >
  <!--ELEMENT projectstatistics EMPTY-->
  <!--ATTLIST projectstatistics
    asgTotalWork CDATA #IMPLIED
    asgCount CDATA #IMPLIED
    asgStarted CDATA #IMPLIED
    asgFinished CDATA #IMPLIED
    asgOverdue CDATA #IMPLIED
    asgActualWork CDATA #IMPLIED
    asgWorkRem CDATA #IMPLIED
    asgOverdueWork CDATA #IMPLIED
    asgRefWork CDATA #IMPLIED
    asgEarnedWork CDATA #IMPLIED
    keyEventCount CDATA #IMPLIED
    keyEventFinishedCount CDATA #IMPLIED
    asgLatestFinish CDATA #IMPLIED
    targetCost CDATA #IMPLIED
    targetFinishDate CDATA #IMPLIED
    targetEffort CDATA #IMPLIED
    issuesOverdueCount CDATA #IMPLIED
    issuesOpenCount CDATA #IMPLIED
    reportTime CDATA #IMPLIED
    projectStartDate CDATA #IMPLIED
    projectFinishDate CDATA #IMPLIED
    refStartDate CDATA #IMPLIED
    refFinishDate CDATA #IMPLIED
    refWork CDATA #IMPLIED
    refCost CDATA #IMPLIED
    outputCount CDATA #IMPLIED
```

```
outputFinishedCount CDATA #IMPLIED
actualCost CDATA #IMPLIED
totalCost CDATA #IMPLIED
>
1>
```

Appendix B. Project Amendment Document Type Definition

```
<?xml version="1.0"?>
<!DOCTYPE PROJECTAMENDMENT [
<!-- document type definition copyright (c) 1998 by Marin Research -->
<!ENTITY lt "&#38;#60;">
<!ENTITY gt "&#62;">
<!ENTITY amp "&#38;#38;">
<!ENTITY apos "&#39;">
<!ENTITY quot "&#34;">
<!ENTITY lf "&#10;">
<!ENTITY cr "&#13;">
<!ELEMENT PROJECTAMENDMENT (projectstatistics)*>
<!ATTLIST PROJECTAMENDMENT
  id CDATA #REQUIRED
  name CDATA #IMPLIED
  reportTime CDATA #IMPLIED
  sourceRepositoryTitle CDATA #IMPLIED
  sourceRepositoryFile CDATA #IMPLIED
  sourceRepositoryServer CDATA #IMPLIED
  programName CDATA #IMPLIED
  managerSendTo CDATA #IMPLIED
  senderType CDATA #IMPLIED
  senderVersion CDATA #IMPLIED
  sponsor CDATA #IMPLIED
  senderEnvironment CDATA #IMPLIED
  zone CDATA #IMPLIED
>
<!ELEMENT projectstatistics EMPTY>
<!ATTLIST projectstatistics
  asgTotalWork CDATA #IMPLIED
  asgCount CDATA #IMPLIED
  asgStarted CDATA #IMPLIED
  asgFinished CDATA #IMPLIED
  asgOverdue CDATA #IMPLIED
  asgActualWork CDATA #IMPLIED
  asgWorkRem CDATA #IMPLIED
  asgOverdueWork CDATA #IMPLIED
  asgRefWork CDATA #IMPLIED
  asgEarnedWork CDATA #IMPLIED
  keyEventCount CDATA #IMPLIED
  keyEventFinishedCount CDATA #IMPLIED
  asgLatestFinish CDATA #IMPLIED
  targetCost CDATA #IMPLIED
  targetFinishDate CDATA #IMPLIED
  targetEffort CDATA #IMPLIED
  issuesOverdueCount CDATA #IMPLIED
  issuesOpenCount CDATA #IMPLIED
  reportTime CDATA #IMPLIED
  projectStartDate CDATA #IMPLIED
  projectFinishDate CDATA #IMPLIED
  refStartDate CDATA #IMPLIED
  refFinishDate CDATA #IMPLIED
  refWork CDATA #IMPLIED
  refCost CDATA #IMPLIED
  outputCount CDATA #IMPLIED
  outputFinishedCount CDATA #IMPLIED
```

```
actualCost CDATA #IMPLIED
totalCost CDATA #IMPLIED
>
|>
```

For more information, please call or visit our web site

Marin Research, Inc.

100 Larkspur Landing Circle, Suite 114

Larkspur, California 94939

info@marinres.com

<http://www.marinres.com>