

Genesis is a methodology and toolset that is developed by Item Solutions. The approach is most probably unique in the world.

Over the last few years the 'Agile'-approach has gained a growing number of supporters and increasing success. The Genesis methodology is based on these agile methods and practices ... but adds a few extras.

Genesis bridges the gap between the functional design and the code base. Up until now, it was assumed that only very formal processes and strict modeling could guarantee a serious level of control over the development process. Genesis proves the contrary. With a minimum of formalism it achieves a perfect link between functional design and code.

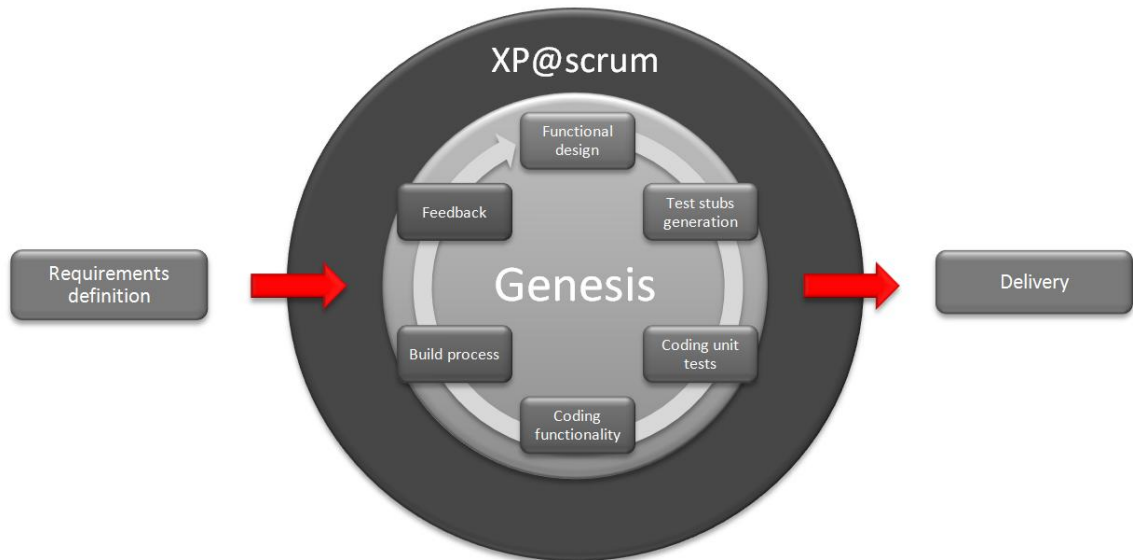
The most difficult aspect of developing software is controlling the entire development process. This means streamlining a multitude of different sub processes; communication, creativity, project management ...

But Genesis functions as an integrating methodology; using Genesis implies control and measurability. Furthermore it is a facilitating methodology as opposed to an invasive one. It does not impose limitations on project teams or its members, instead it supports and facilitates repetitive tasks.

Let's see how it works

1 Genesis workflow

The Genesis workflow can be graphically represented as follows:



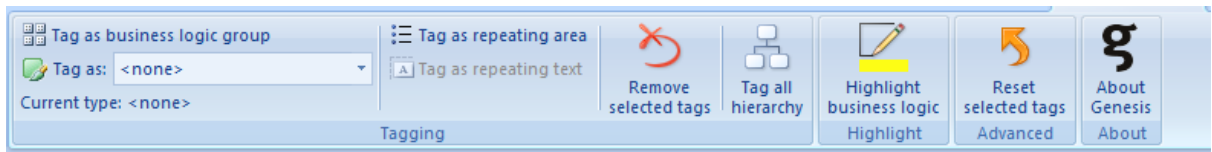
You can see that Genesis integrates several practices from the agile community such as:

- Scrum@XP
- Test Driven Development
- Continuous integration
- Refactoring
- ...

2 Genesis Functionality

2.1 Identifying business logic

When writing the functional design, all business logic is identified and ‘tagged’. This tagging process actually means ‘making the functional design Genesis compliant’.



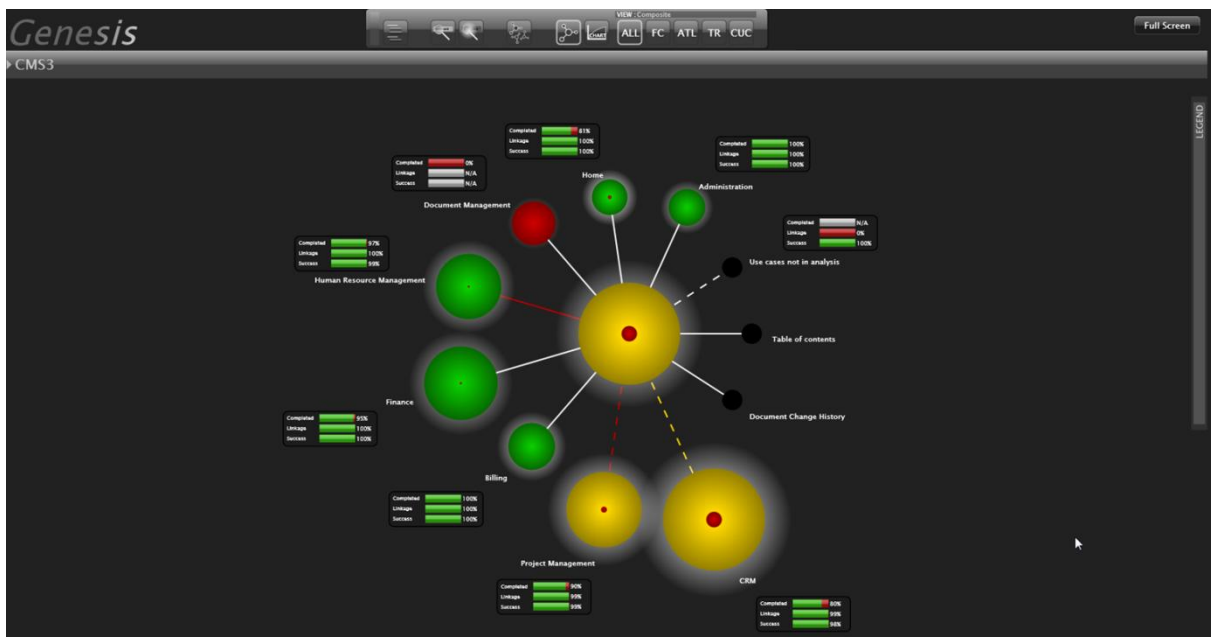
The Genesis Tagger is used by the analyst to indicate what business logic needs to be tracked by Genesis.

2.2 Project management

The Genesis Silverlight Client provides all team members with a crystal clear overview of a software project. At this moment, following functionalities are available:

2.2.1 Project overview

This overview clearly indicates the status of a project. The figure below is a representation of the current status of a development project of some 15.000 hours.



Following things can be said about the current status:

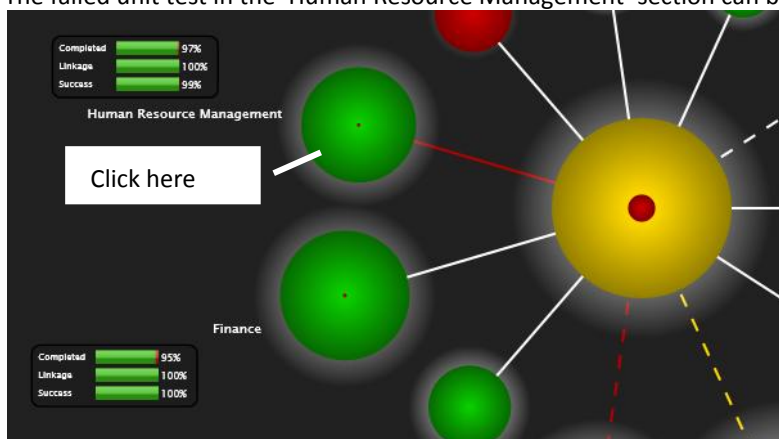
- The functional design needs to be adapted in the sections ‘Project Management’ and ‘CRM’
- Development has not started on ‘Document Management’
- In the ‘Project Management’ section, at least 1 unit test has failed. Also, at least 1 unit test is erroneously linked to business logic
- In the ‘Human Resource Management’ section, at least 1 unit test has failed.
- For the ‘Finance’ section, a little bit more functionality needs to be developed
- Same for ‘Project Management’ ...
- ... and for ‘CRM’

So we can take a closer look ...

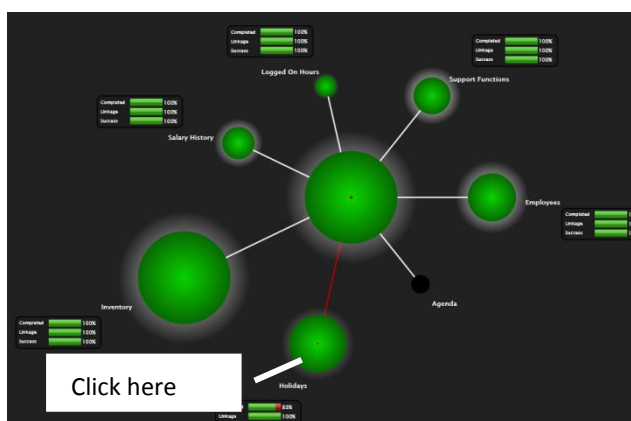
2.2.2 Drill down functionality

Through ‘drilling down’ we can look at the whole application :

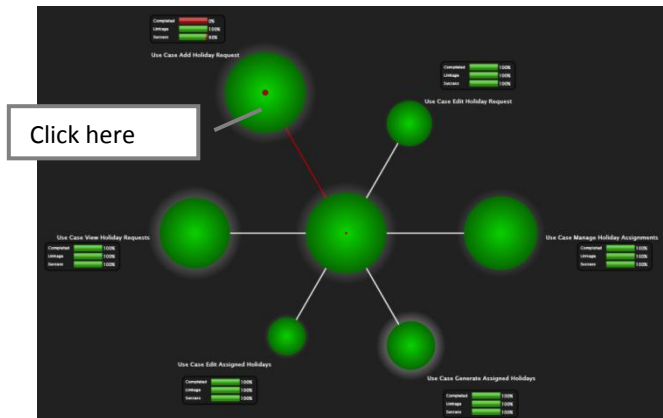
The failed unit test in the ‘Human Resource Management’ section can be quickly found



Just click on ‘Human Resource Management’...



... and we immediately notice that the error is produced in the ‘Holidays’ section ... so we click ...



... and click once more on 'Use Case Add Holiday Request' because that is where the red line appears

1 level deeper and we can look at the use case itself.

Preconditions

- The user has business operation `HumanResourceManagement_RequestHolidays`.
- `...Holidays.UseCaseAddHolidayRequest_TestSubmitHolidayRequest_WithoutAnyAuthorization_SecurityException`
- `...ent.Holidays.UseCaseAddHolidayRequest_TestSubmitHolidayRequest_WithAuthorization_NoSecurityException`
- `...dHolidayRequest_TestSubmitHolidayRequest_WithAuthorizationInRelatedBusinessUnit_SecurityException`

Normal Flow

- The system shows a dropdown with employees visible to the current user, by default the users own employee is selected. If the user has business operation `HumanResourceManagement_ViewHolidays` or `HumanResourceMangement_ManageHolidays`, he/she can select his/her own name and the ones he/she is superior of.
- If the user has business operation `HumanResourceManagement_ViewAllHolidays` or `HumanResourceManagement_ManageAllHolidays`, he/she can see all employees that are in a business unit where the user has these business operations for.
- The user can add 1 or more holidays to the request by completing following fields and clicking the "Add Holiday" button:
 - Field: Optional; Type: Data; Validation
 - Holiday Date: N; Date Control; [Default value: Current date]; Valid Date
 - Holiday Time: N; Radio Button; [Values: "All Day", "AM", "PM"]
 - [Default: "All Day"]; Limit to list
 - Holiday Types: N; Drop Down; [Data: every holiday type that is assigned to the user]; Limit to list
 - Employee: N; Person Drop down; See above; Limit to list
 - Number of days: N; Textbox
 - Valid number
 - Comment: Y; Textbox
 - Length of field
- `...CMS.Tests.CrudTests.HumanResourceManagement.TestHolidayDALC.TestPersist_NewHoliday_HolidayInserted`
- The system shows an overview of all requested days
[Data: Employee, Date, Start Time, End Time, Holiday Type, Comment]
- If the user requests more than 1 number of days and the start date is not in a weekend, the system will automatically pick the next day available that is not in a weekend.

7.3.5 Use Case Add Holiday Request

7.3.5.1 Preconditions

- The user has business operation `HumanResourceManagement_RequestHolidays`.

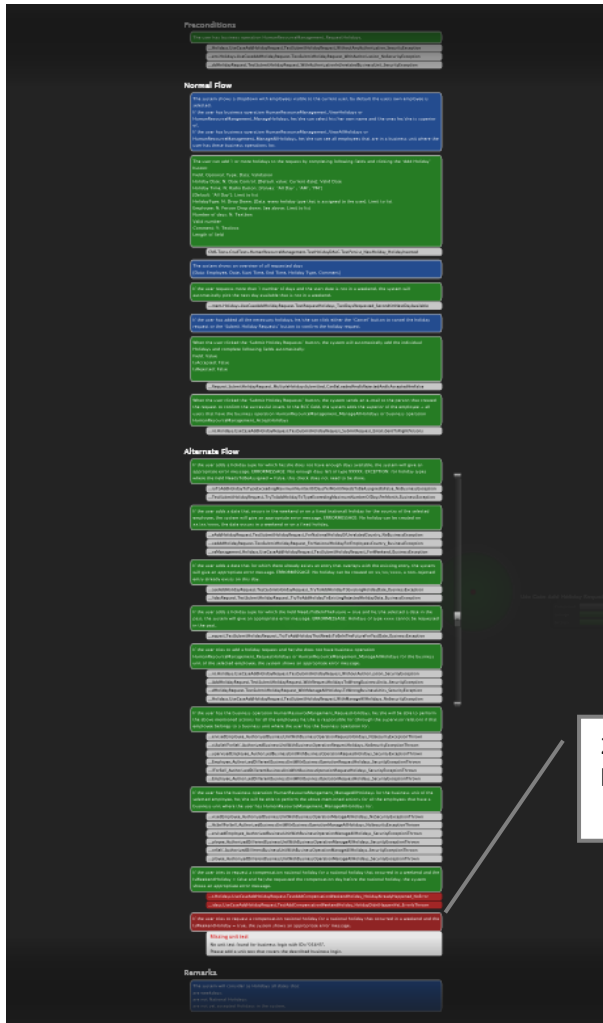
7.3.5.2 Normal Flow

- The system shows a dropdown with employees visible to the current user, by default the users own employee is selected.
 - If the user has business operation `HumanResourceManagement_ViewHolidays` or `HumanResourceMangement_ManageHolidays`, he/she can select his/her own name and the ones he/she is superior of.
 - If the user has business operation `HumanResourceManagement_ViewAllHolidays` or `HumanResourceManagement_ManageAllHolidays`, he/she can see all employees that are in a business unit where the user has these business operations for.
- The user can add 1 or more holidays to the request by completing following fields and clicking the "Add Holiday" button:

Field	Optional	Type	Data	Validation
Holiday Date	N	Date Control	[Default value: Current date]	Valid Date
Holiday Time	N	Radio Button	[Values: "All Day", "AM", "PM"] [Default: "All Day"]	Limit to list
HolidayType	N	Drop Down	[Data: every holiday type that is assigned to the user]	Limit to list
Employee	N	Person Drop down	See above	Limit to list
Number of days	N	Textbox		Valid number
Comment	Y	Textbox		Length of field

- The system shows an overview of all requested days
[Data: Employee, Date, Start Time, End Time, Holiday Type, Comment]
- If the user requests more than 1 number of days and the start date is not in a weekend, the system will automatically pick the next day available that is not in a weekend.
- If the user has added all the necessary holidays, he/she can click either the "Cancel" button to cancel the holiday request or the "Submit Holiday Requests" button to confirm the holiday request.

The use case is rather large so we scroll down ...



2 failed unit tests + 1 piece of uncovered business logic

... and here they are ...

...AsSelfForSelf_AuthorizedBusinessUnitWithBusinessOperationManageAllHolidays_NoSecurityExceptionThrown

...evisedEmployee_AuthorizedBusinessUnitWithBusinessOperationManageAllHolidays_SecurityExceptionThrown

...mployee_AuthorizedDifferentBusinessUnitWithBusinessOperationManageAllHolidays_SecurityExceptionThrown

...orSelf_AuthorizedDifferentBusinessUnitWithBusinessOperationManageAllHolidays_SecurityExceptionThrown

...mployee_AuthorizedDifferentBusinessUnitWithBusinessOperationManageAllHolidays_SecurityExceptionThrown

If the user tries to request a compensation national holiday for a national holiday that occurred in a weekend and the IsWeekendHoliday = false and he/she requested the compensation day before the national holiday, the system shows an appropriate error message.

...t.Holidays.UseCaseAddHolidayRequest.TestAddCompensationWeekendHoliday_HolidayAlreadyHappened_NoError

...idays.UseCaseAddHolidayRequest.TestAddCompensationWeekendHoliday_HolidayDidntHappenYet_ErrorsIsThrown

If the user tries to request a compensation national holiday for a national holiday that occurred in a weekend and the IsWeekendHoliday = true, the system shows an appropriate error message.

Missing unit test

No unit test found for business logic with ID="03845".
Please add a unit test that covers the described business logic.

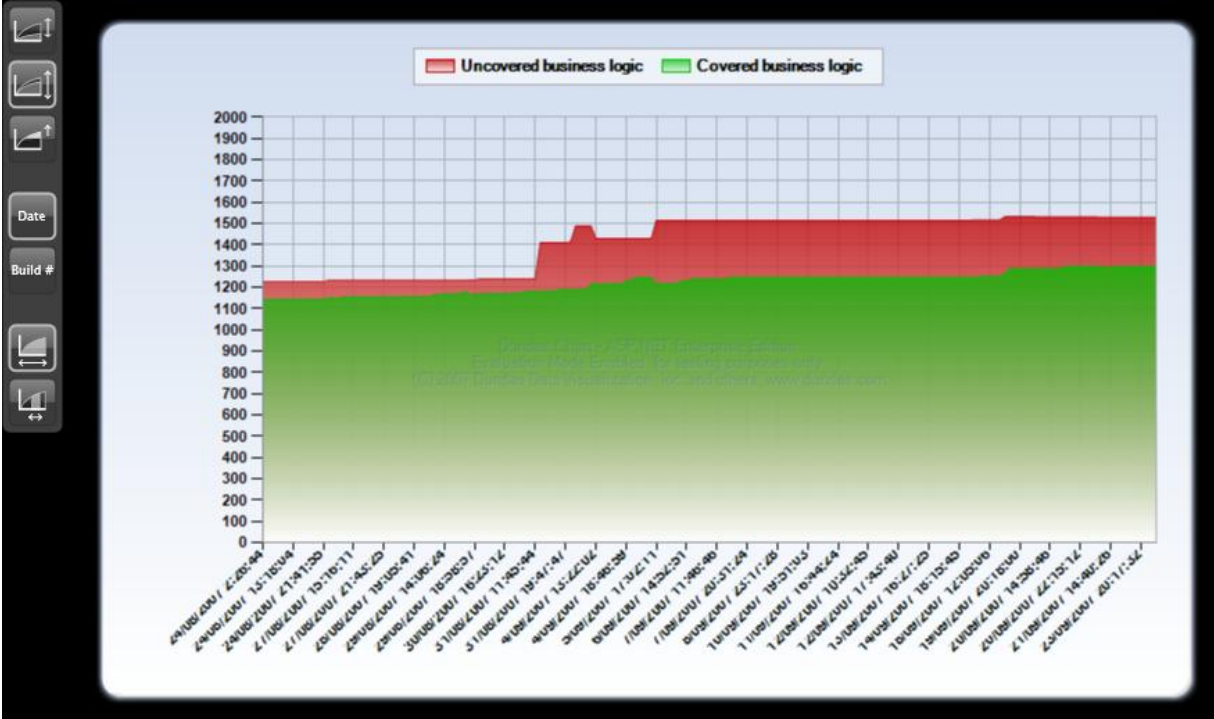
It is clear that 2 unit tests have failed. Furthermore the block of business functionality at the bottom is not covered by tests. This use case is therefore 'not complete' (so a bit more coding to do).

2.2.3 Graphical representation of project progress

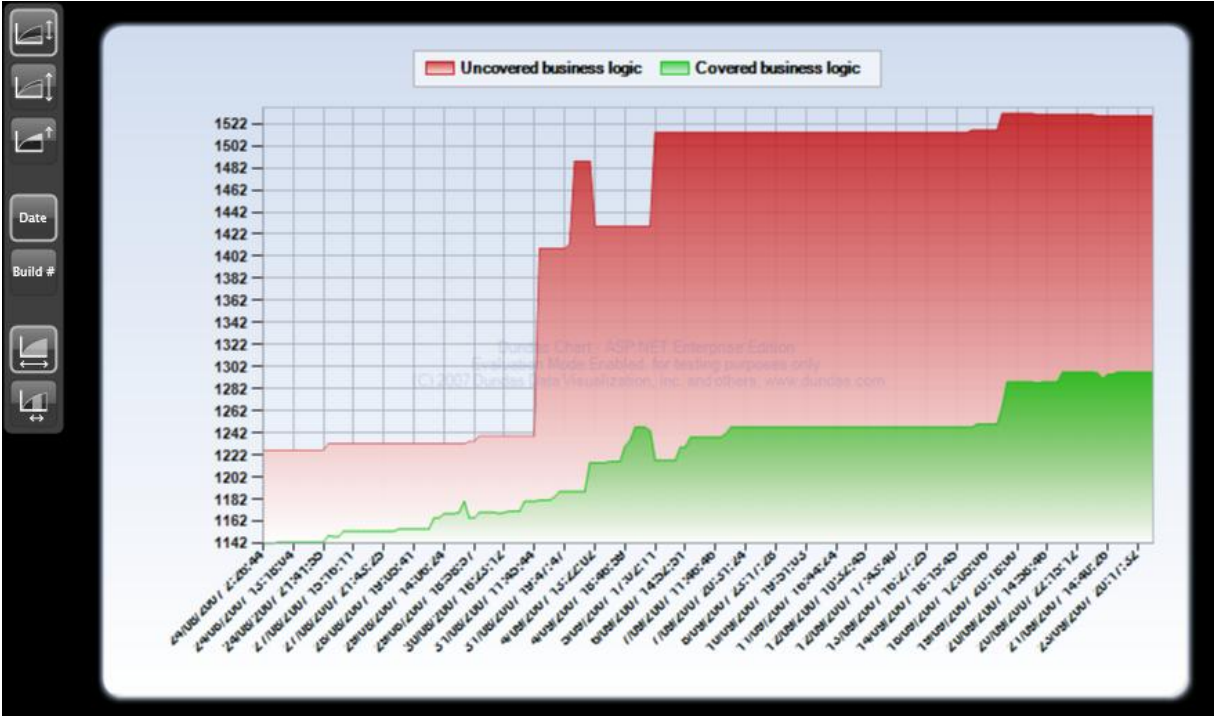
Genesis offers a very clear overview of project progress. Lots of graphs can be produced such as ...

2.2.3.1 Business logic coverage

This graph clearly indicates what business logic is covered by unit tests.

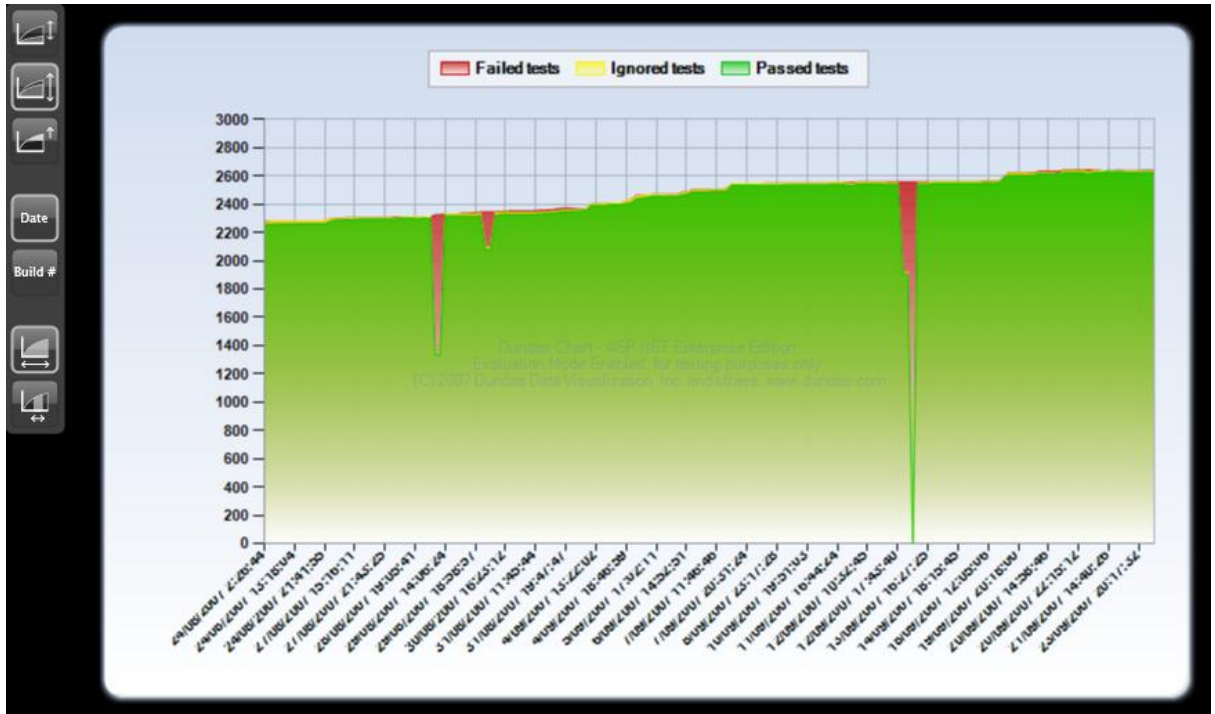


Only top values ...

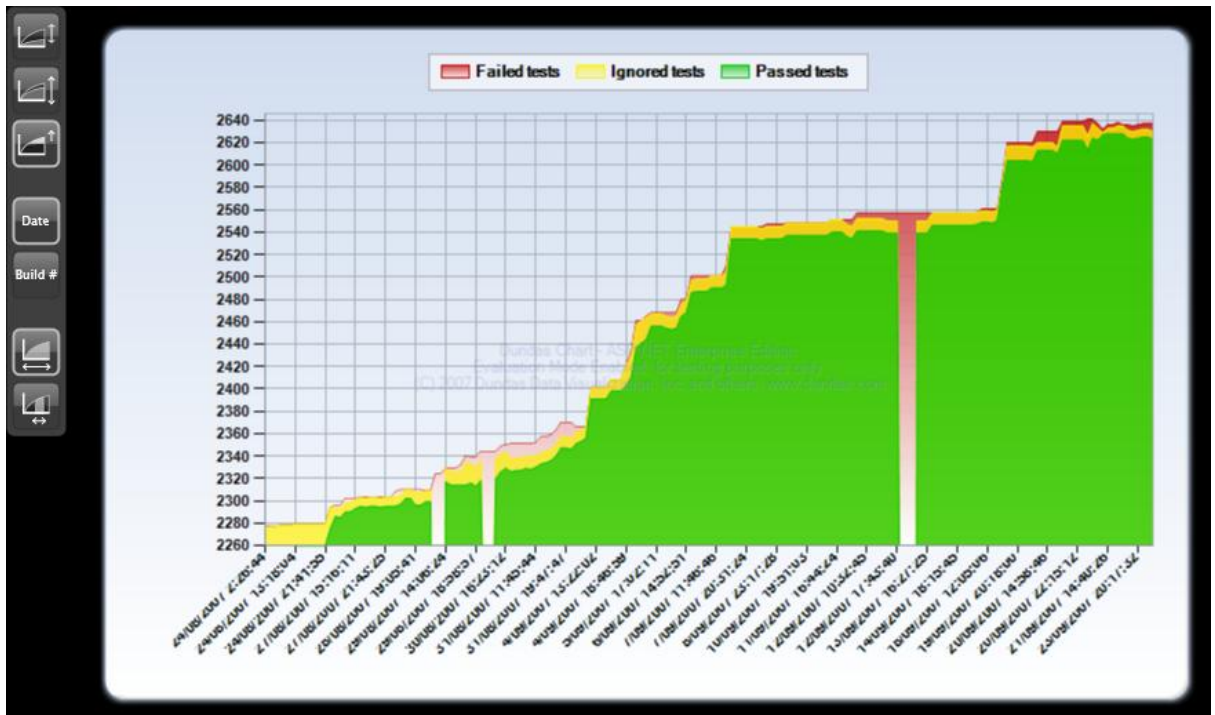


2.2.3.2 Acceptance test history

This graph shows the result of the acceptance tests.

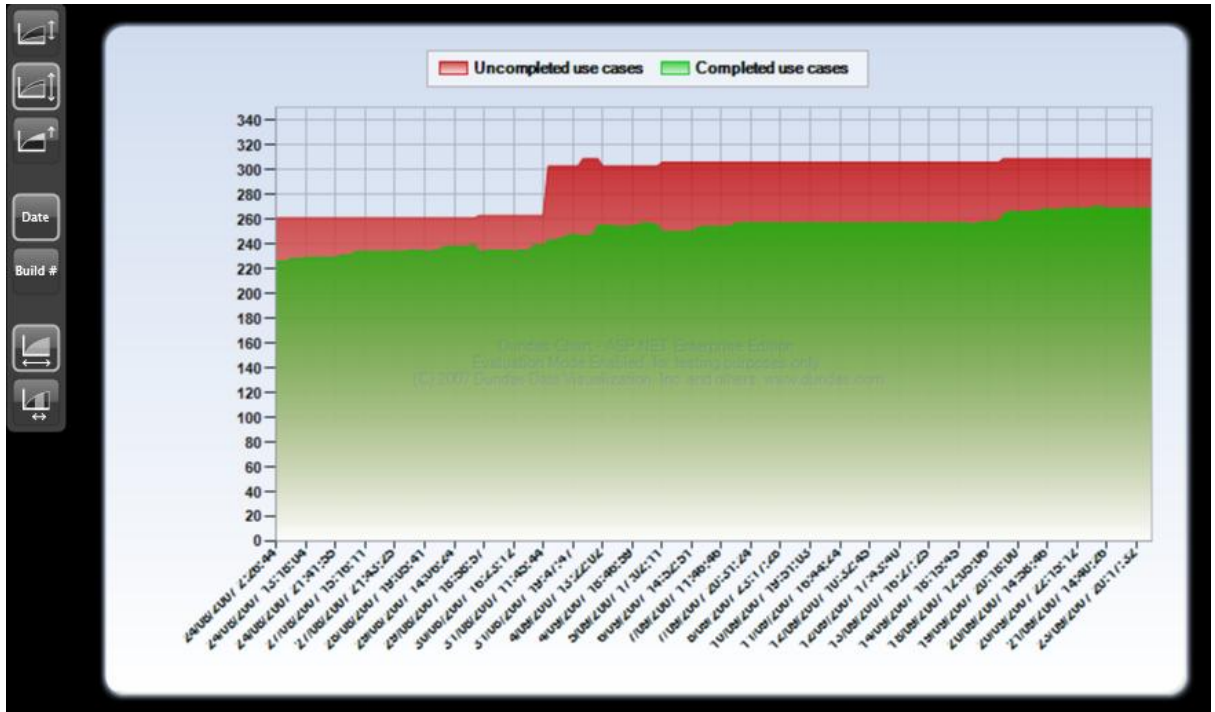


Only top values ...

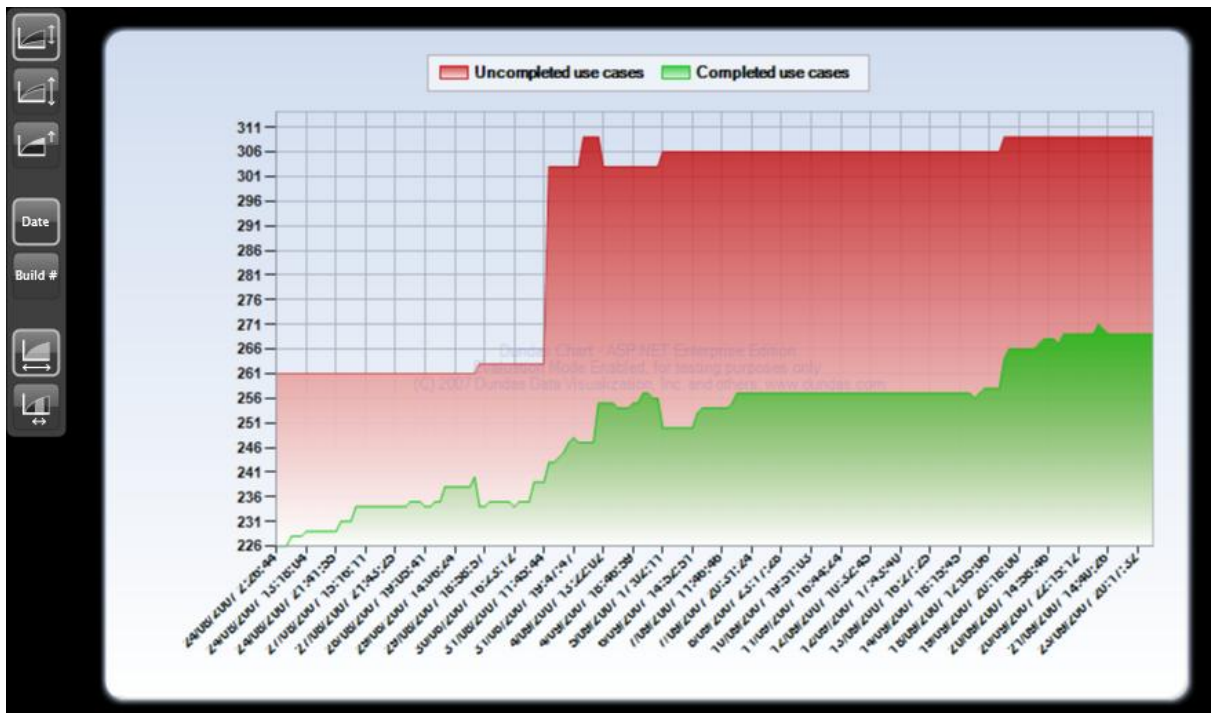


2.2.3.3 Completed use case history

This graph shows the evolution of the completed blocks of functionality (use cases).



Only top values ...



2.2.4 Lists

Several ad hoc lists can be immediately consulted ...

2.2.4.1 'Comments'-report

All communications between developers and analysts are shown.



The screenshot displays a 'Comments'-report for various use cases. Each entry includes a use case title, a category (e.g., Business logic, Alternate Flow, Preconditions), and a description of the logic or flow. Comments from analysts and developers are shown in yellow speech bubbles, often pointing to specific parts of the logic or flow descriptions.

- Use Case Lead Qualification** (Business logic):
 - Comment: "I think this is required to preserve consistency in the database (ATO)"
 - Logic: "If the user tries to create a Lead Qualification for a Step that belongs to a different Scheme than the selected Lead, the system shows an appropriate error message." (Missing business logic)
- Use Case Create Mail Sales Task** (Alternate Flow):
 - Comment: "Implemented in the web layer, because this situation would cause a security exception most of the times and that would hide the business exception of the validation"
 - Flow: "If not a single contact or company is an addressee (only employees), the system will give an appropriate error message." (Missing business logic)
- Use Case View / Edit Sales Task** (Alternate Flow):
 - Comment: "This is not a requirement anymore"
 - Flow: "If a sales task is completed and the user tries to edit the sales task, the system shows an appropriate error message." (Missing business logic)
- Use Case Select Employee / Contact / Company For Sales Task** (Preconditions):
 - Comment: "Authorization is implemented in the Persist Sales Task command"
 - Precondition: "The user has business operation CRM_ManageSalesTasks for on of the business units of the selected company"
- Use Case Select Employee / Contact / Company For Sales Task** (Alternate Flow):
 - Comment: "Not testable - implemented on the web layer"
 - Flow: "The system only shows those companies, contacts who are visible to the user." (Missing business logic)
- Use Case View / Edit Qualification Stage** (Business logic):
 - Comment: "I think this is needed (ATO)"
 - Logic: "If the user tries to change the Qualification Scheme of a Qualification Stage that is already in use by Lead Qualifications, the system shows an appropriate error message." (Missing business logic)
- Use Case Edit Qualification Step** (Business logic):
 - Comment: "I think this is needed (ATO)"
 - Logic: "If the user tries to change the Qualification Stage of a Qualification Step that is already in use by Lead Qualifications, the system shows an appropriate error message." (Missing business logic)
- Use Case Project Priorities** (Business logic):
 - Logic: "When a lot of transactions are made, the priorities of costcenters with the same parent should be updated accordingly" (Missing business logic)
 - Comment: "I've add a test with a lot of transactions to test the methods that have something to do with 'move priority', and they work.."

2.2.4.2 'Untestable'-report

This list shows all business logic, for which the development team considers the functionality as impossible to cover with unit tests. This report forms the basis of testplan, so that automated tests are completed with manual tests. This part of Genesis is currently being extended. The idea is to ingrate test plan management with project and operation meta-data so maximum control over application quality is attained.



Use Case Manage Purchase Orders

Normal Flow

- The system shows an overview of all PO's
[PONumber, Description, Amount, Currency.Description, Assigned Amount (Amount that is already assigned to a cost) Business Unit, RequestDate, Requester, ApprovalData, Approver, IsAccepted/IsRejected(true/false Icon)]
[Default: All open PO's of the current month and current year]
[SortOrder: PONumber Asc]
- The user can accept/reject purchase orders by selecting the purchase orders and clicking the accept/reject button. The system navigates to a new page (see Use Case Accept/Reject Purchase Order) Remark: If the user does not have business operation Finance_ApprovePurchaseOrder for any business unit, the system hides the "Accept PO" and "Reject PO" button
- The user can add a new Purchase Order Request by clicking the "Add PO Request" button. The user is redirected to a new page (see Request Purchase Order). Remark: If the user does not have business operation Finance_RequestPurchaseOrders for any business unit, the system hides the "Add PO Request" button.
- The user can edit a purchase order (see Use Case Edit Purchase Order) by clicking on the concerning row.
- The user can link one or more costs to the purchase order (see Use Case Link Costs To Purchase Orders)

Use Case Edit Purchase Order

Preconditions

- The user has selected a purchase order in the PO overview (see Use Case Manage Purchase Orders)

Normal Flow

- The system display the details of the selected Purchase Order.
Field; Optional; Type; Data; Validation
Business Unit; N; BU dropdown; [Values: Select one + All business unit where the current user has business operation Finance_ManagePurchaseOrders for]; Limit to list
PONumber; Y; Label
Valid PONumber
Description; N; Text Area
Valid Length
Comment; Y; Text box
Valid length
Amount; N; Text Box
Valid decimal
Currency; N; Dropdown; [Values: All Currencies available]; Limit to list
Current Euro Rate; N; Label; [Current EuroRate of the selected currency]
Request Date; N; Label
Valid Date
Requester; N; Label

Approval Date; Y; Label

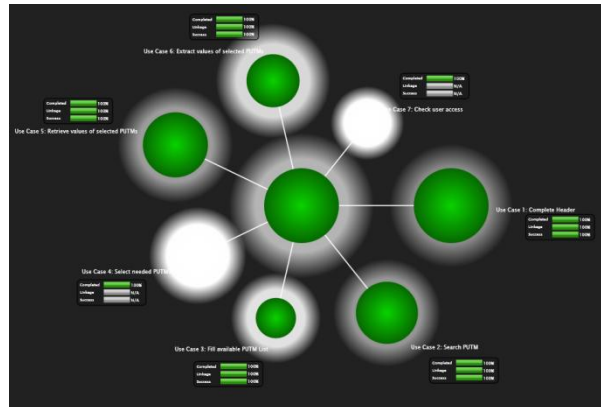
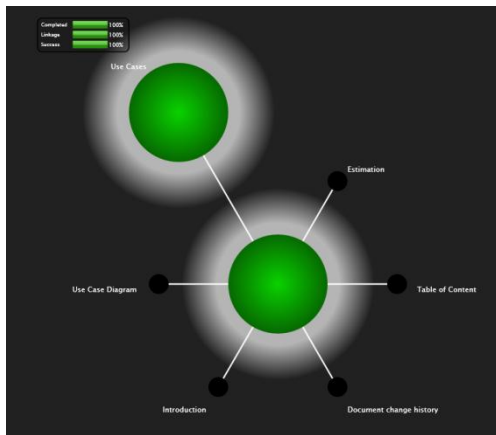
Approver; Y; Label; [Value: Current Application User]
; Accepted; N; Image; [Values: Accepted/Rejected Image]
- Underneath the details of the current Purchase order the user will be able to accept/reject the purchase order (see Use Case Accept/Reject Purchase Order). After the user has accepted/rejected the PO, the system updates the fields in the details view.

2.2.5 Summary

Using Genesis leads to faultless and correct completion of development projects.

A small example of the process ...

2.2.5.1 Overview



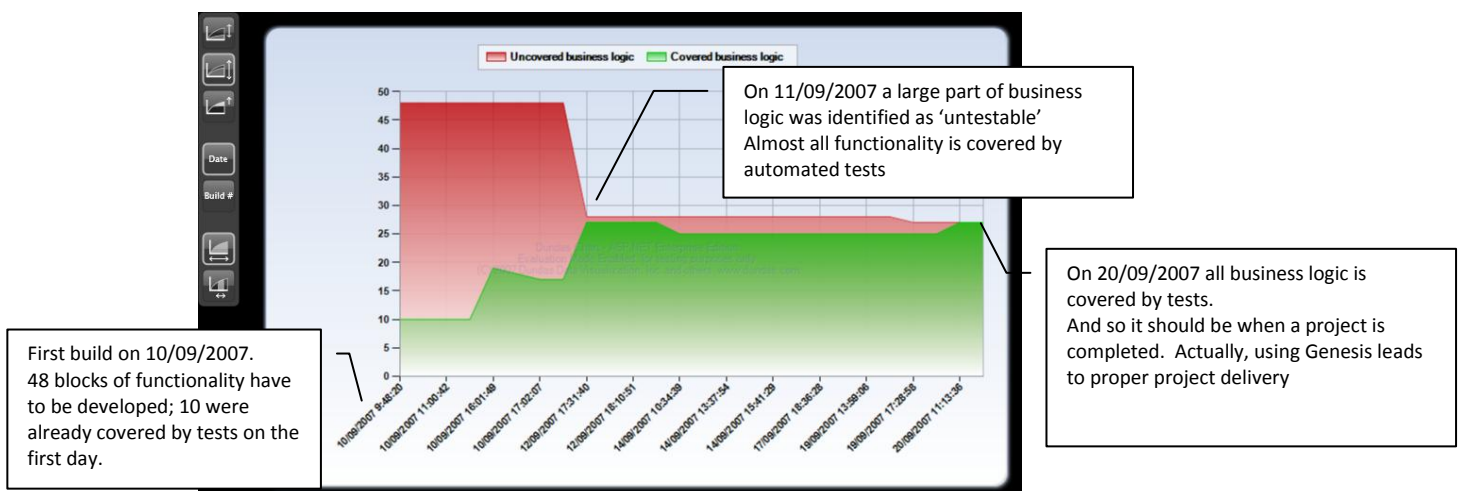
All tests have clearly succeeded (white lines)

Alle testen zijn gelukt (witte lijnen) en alle functionaliteit is ontwikkeld (enkel groene cirkels). Er is weliswaar vrij veel 'untestable' business logica (de witte 'nevel' rond de cirkels).

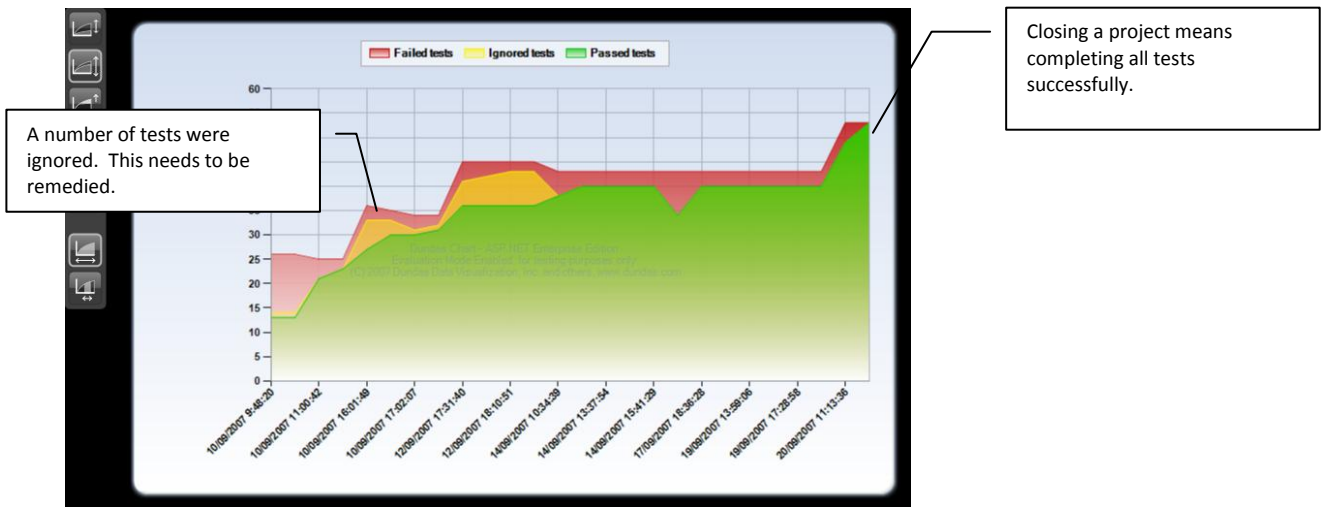
2.2.5.2 Graphs

The resume of this little project progress.

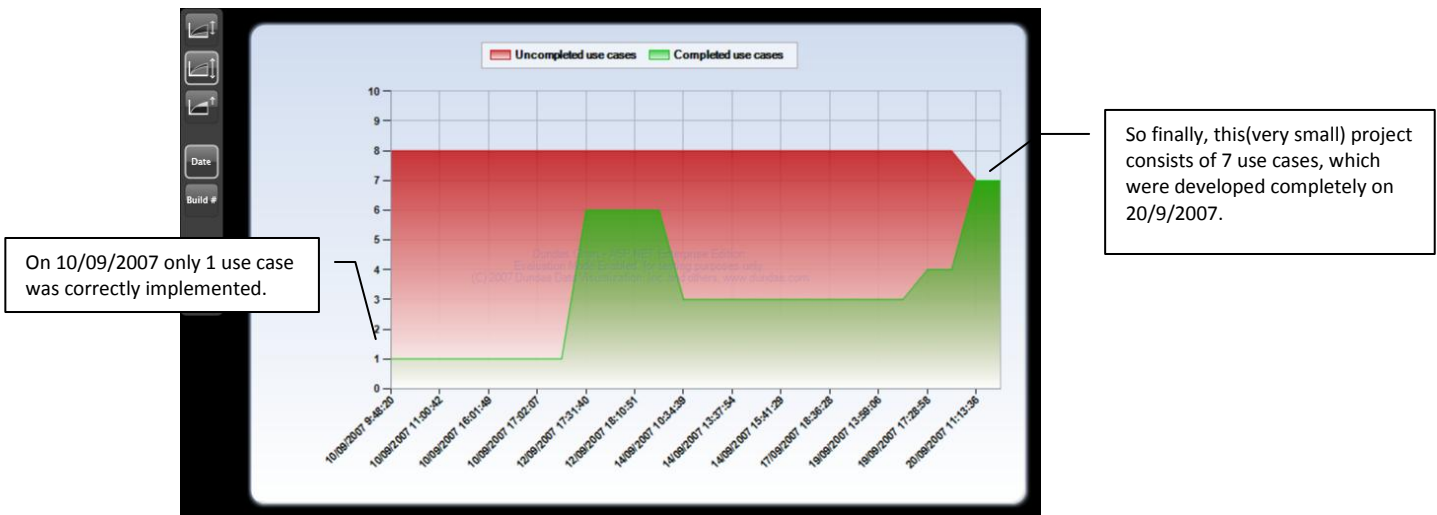
Business logic coverage ...



History of the acceptance tests ...



Use case history ...



The linkage of the code with the functional design will be clear in the next paragraph.

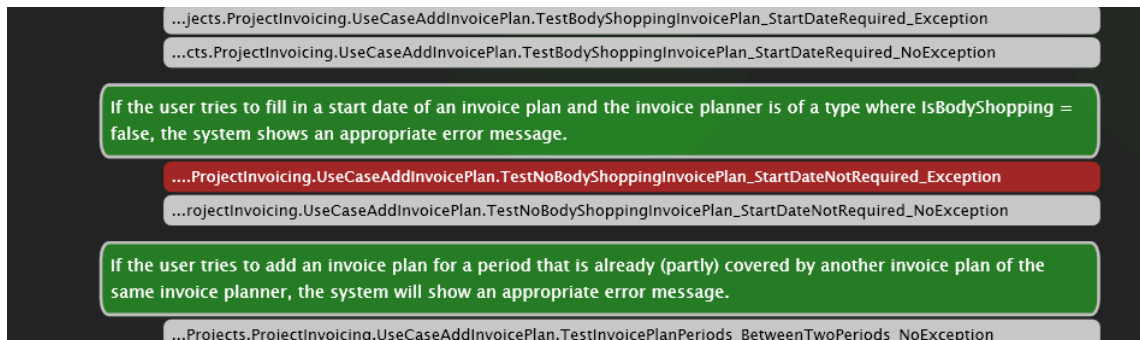
2.3 Genesis Client Tools

The Client Tools integrate Genesis with the development environment. This environment is not limited to 1 technology. We currently have Genesis available for .Net and Java development.

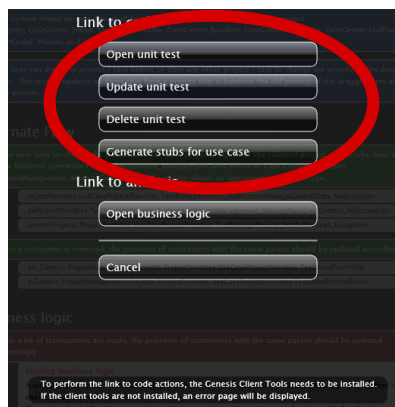
Following functionality is offered by the Client Tools:

2.3.1 Link to code

Navigate to a unit tests with an error (so follow the red line)...

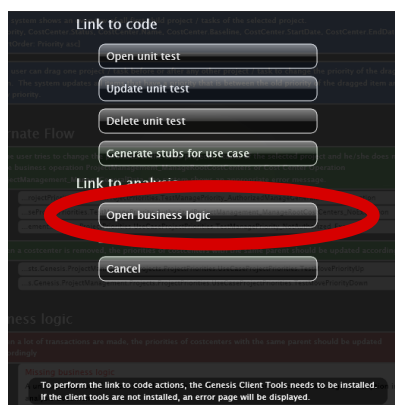


And the test can immediately be localized in the codebase.



2.3.2 Link to functional design

Inversely, the functional design can be immediately adapted to synch code and design.



2.4 Genesis build server

The Genesis Build Server keeps the functional design synchronized with the codebase. By using continuous integration the slightest difference between analysis and code is immediately detected.

This synchronization gets clear when we compare the text of the MS Word document with the text in the Genesis Database.

6.3 Purchase Orders

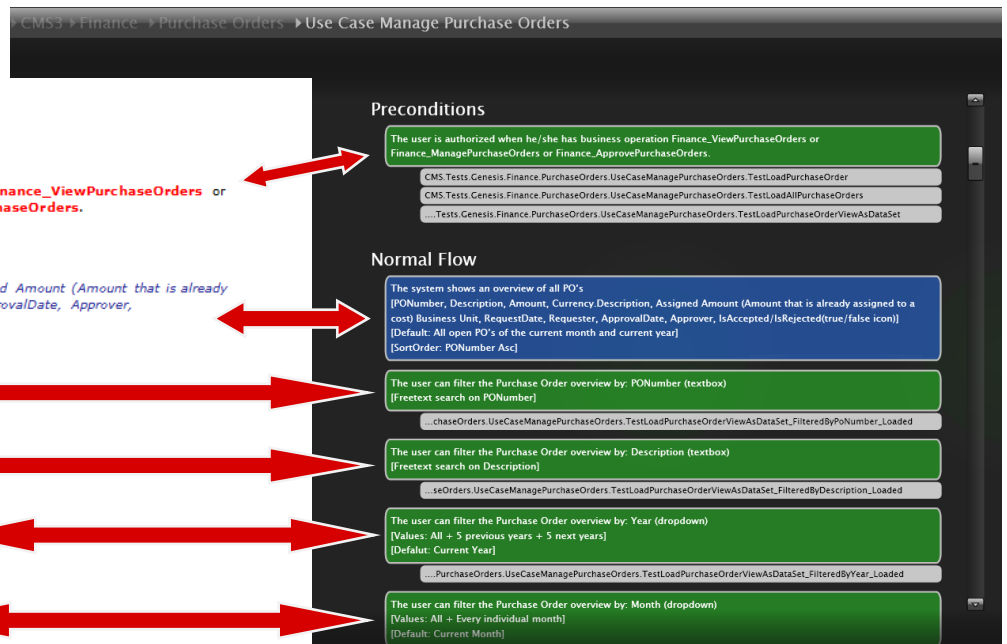
6.3.1 Use Case Manage Purchase Orders

6.3.1.1 Preconditions

- The user is authorized when he/she has **business operation Finance_ViewPurchaseOrders** or **Finance_ManagePurchaseOrders** or **Finance_ApprovePurchaseOrders**.

6.3.1.2 Normal Flow

- The system shows an overview of all PO's
[PONumber, Description, Amount, Currency.Description, Assigned Amount (Amount that is already assigned to a cost) Business Unit, RequestDate, Requester, ApprovalDate, Approver, IsAccepted/IsRejected(true/false icon)]
[Default: All open PO's of the current month and current year]
[SortOrder: PONumber Asc]
- The user can filter the Purchase Order overview by:
 - PONumber (textbox)
[Freetext search on PONumber]
 - Description (textbox)
[Freetext search on Description]
 - Year (dropdown)
[Values: All + 5 previous years + 5 next years]
[Default: Current Year]
 - Month (dropdown)
[Values: All + Every individual month]
[Default: Current Month]
 - Currency (dropdown)
[Values: All + every individual currency]
[Default: All]
 - Requester (dropdown)
[Values: All + every individual employee that is active in the system (Person.IsActive)]
 - Business Unit (Business Unit Dropdown)
[Values: All + All Business Unit where the user has business operation Finance_ViewPurchaseOrders or Finance_ManagePurchaseOrders for.]
[Default: Business Unit selected on the homepage]
 - PO status (dropdown)
[Values: All + Open (all PO's that are not yet fully assigned to one or more cost(s)) + Closed (all PO's that are fully assigned to one or more costs.)]



Preconditions

- The user is authorized when he/she has business operation Finance_ViewPurchaseOrders or Finance_ManagePurchaseOrders or Finance_ApprovePurchaseOrders.
- CMS.Tests.Genesis.Finance.PurchaseOrders.UseCaseManagePurchaseOrders.TestLoadPurchaseOrder
- CMS.Tests.Genesis.Finance.PurchaseOrders.UseCaseManagePurchaseOrders.TestLoadAllPurchaseOrders
- Tests.Genesis.Finance.PurchaseOrders.UseCaseManagePurchaseOrders.TestLoadPurchaseOrderViewAsDataSet

Normal Flow

- The system shows an overview of all PO's
[PONumber, Description, Amount, Currency.Description, Assigned Amount (Amount that is already assigned to a cost) Business Unit, RequestDate, Requester, ApprovalDate, Approver, IsAccepted/IsRejected(true/false icon)]
[Default: All open PO's of the current month and current year]
[SortOrder: PONumber Asc]
- The user can filter the Purchase Order overview by: PONumber (textbox)
[Freetext search on PONumber]
...chaseOrders.UseCaseManagePurchaseOrders.TestLoadPurchaseOrderViewAsDataSet_FilteredByPoNumber_Loaded
- The user can filter the Purchase Order overview by: Description (textbox)
[Freetext search on Description]
...seOrders.UseCaseManagePurchaseOrders.TestLoadPurchaseOrderViewAsDataSet_FilteredByDescription_Loaded
- The user can filter the Purchase Order overview by: Year (dropdown)
[Values: All + 5 previous years + 5 next years]
[Default: Current Year]
...PurchaseOrders.UseCaseManagePurchaseOrders.TestLoadPurchaseOrderViewAsDataSet_FilteredByYear_Loaded
- The user can filter the Purchase Order overview by: Month (dropdown)
[Values: All + Every individual month]
[Default: Current Month]

Of course, continuous integration in itself brings a lot of advantages. Fast feedback allows for easy and rapid detection of bugs. As a consequence, they get fixed very fast.

3 Genesis Topology & toolset

The deceptively easy and intuitive interface of the Genesis tools, hide a very extensive technical and functional model. For reasons of confidentiality we can only describe them summarily.

Genesis facilitate the work of developers, analysts and project managers. Furthermore it offers a clear insight into progress, status and quality of the project to non-technical collaborators.

Genesis consists of some 28 modules that are exposed via the following tools:

- **Genesis tagger**
De MS Word add-in that 'tags' the business logic
- **Genesis reporting engine**
The reporting engine creates the Genesis XML report by comparing the MS Word analysis, the unit test codebase and the Xunit tes report
- **Genesis Silverlight Client**
In order to look at the Genesis results efficiently, the Silverlight Client is used.
- **Genesis Client Tools**
The Genesis Client Tools automate certain tasks such as opeing, updating or generating unit tests. To accomplish this, the client calls specific automation clients such as a code automation client or an analysis automation client
This automation client makes use of the Genesis protocol, which can be called by a browser
- **Genesis Test Generator**
This component generates the 'method stubs' (empty methods or 'test skeleton') for the unit tests
- **Build2Genesis Services**
The Build2Genesis system consists of 2 parts: a command line client and a service. The client application is part of the build system. After every build the client will transmit the build info to the sevice
- **Genesis Intelligence Services**
These services are being used as a frontend for the Genesis information in the database. The service can be used by client-applications such as the Genesis Silverlight client, to collect information about a certain project or build

4 Demo

If you would like to see Genesis in real life request your login credentials at www.itemsolutions.com