SAP Fiori UX Design and Build Challenge

1 The Story

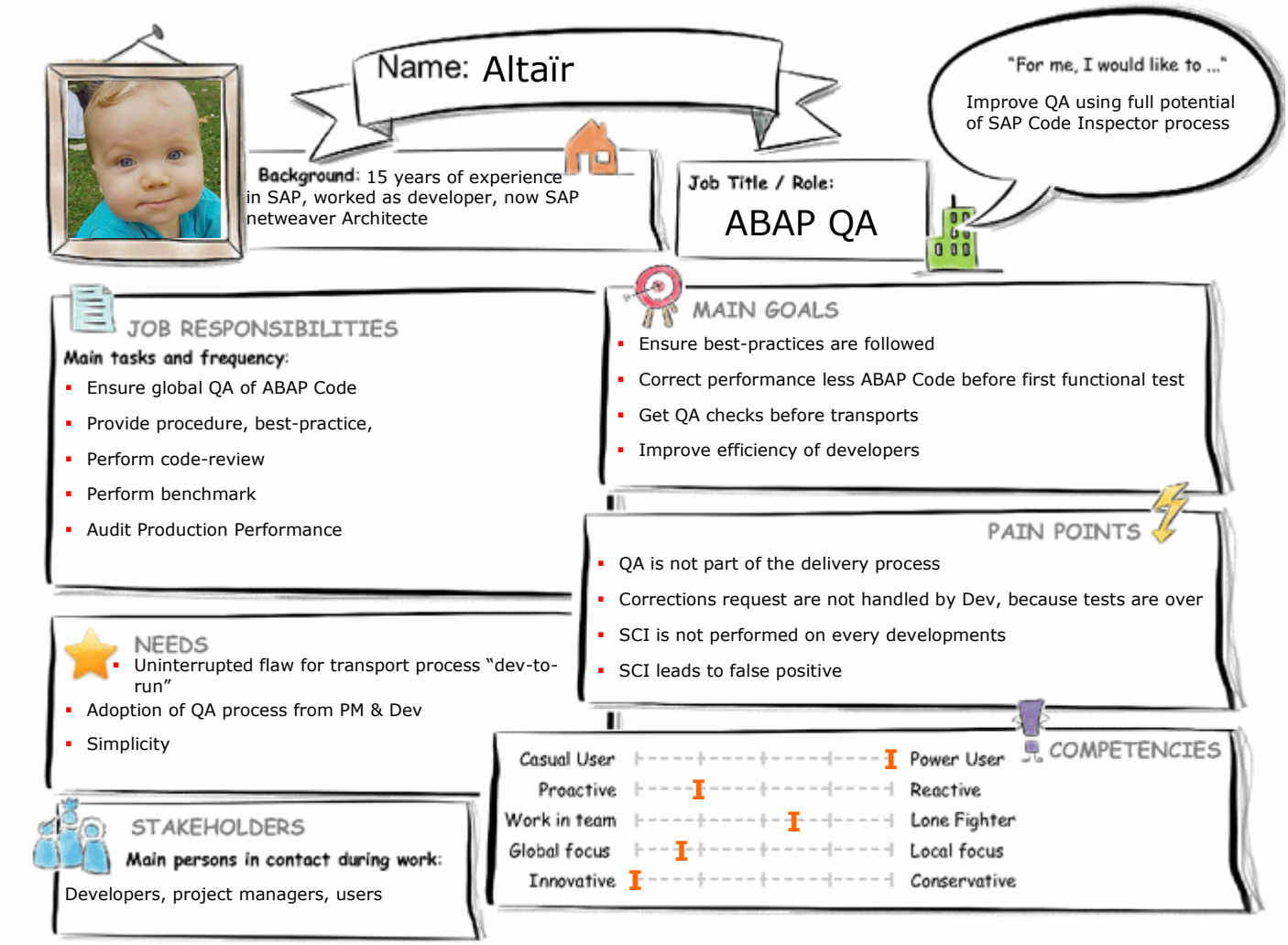
As a developer I do not have so much functional experiences. So I focus on my needs. Within SAP Code Inspector (Transaction SCI) there is an approval process for code inspection errors. Developer who has made SCI run (let's say before releasing a transport request) might ask for an exception. An exception is derogation to an error returned by a SCI control. Under **certain specials** circumstances the error should be ignored. This exception should be **motivated and explained**. To prevent each time an SCI run on the same object (that have an error that should be ignored) an exception request and approval process is provided by SAP on transaction SCI. Persons who ask for exception are from the developer team, and persons who validate or reject the exception request are part of the QA team. This Fiori apps is focused on the validation (or rejection) from the QA team.

Segmentation: All segments that are involved in ABAP QA

Target: QA validator

Positioning: provide more efficient and reactive way to approve or reject exceptions.

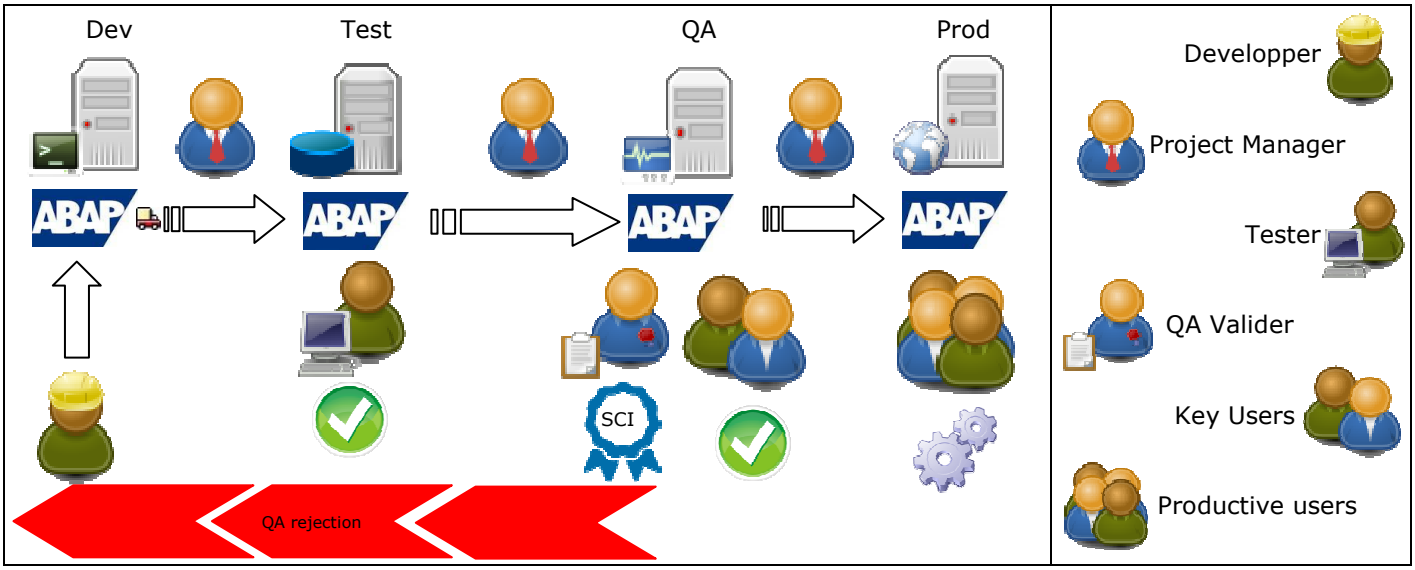
2 The Persona



3 The Processes

3.1 Generalized QA process

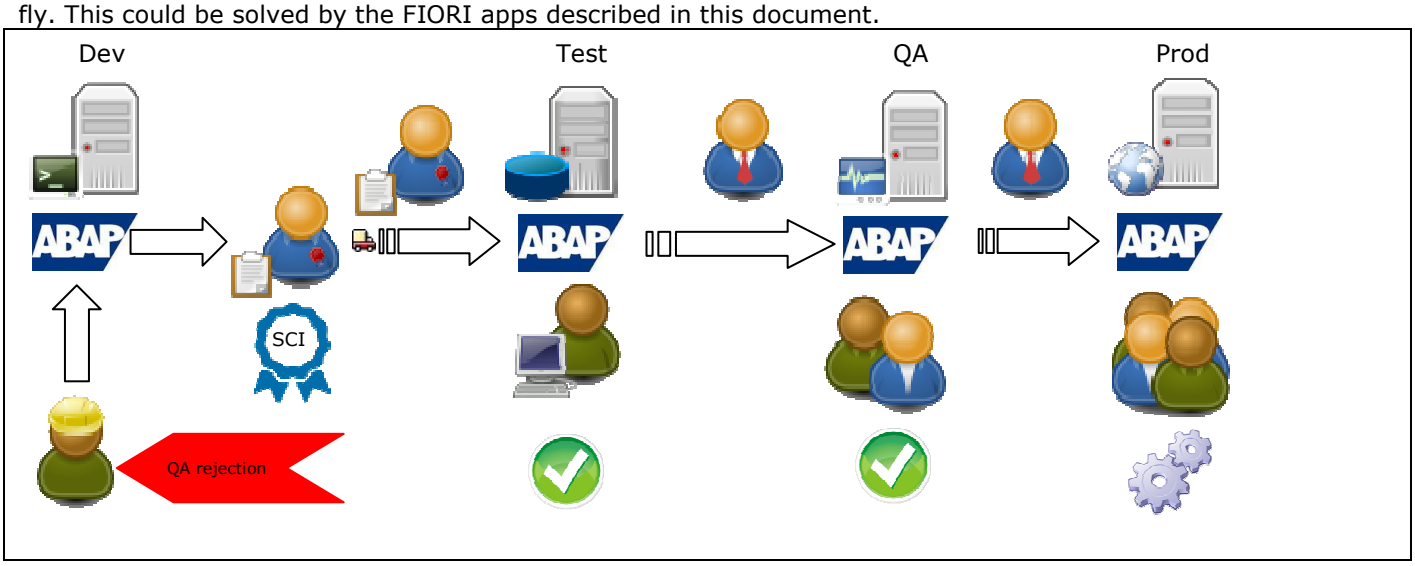
Here is the End-to-End QA process we often view because simpler to organise with 2 distinct phases with 2 distinct teams (and responsibilities):



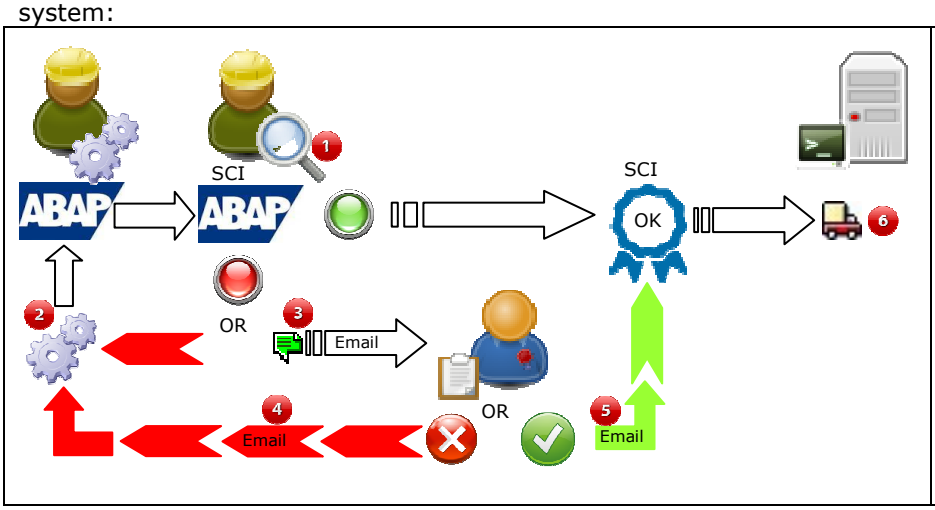
QA remarks, or corrections, are made on QA system and do not always leads to change in Development systems if Key Users have validated the developments.

3.2 Wanted QA process

Find below the End-to-End process that is requested by the QA team to dramatically improve development quality. This process could not be implemented because of the communication overhead that produce the QA validation on the fly. This could be solved by the FIORI apps described in this document.



QA is made before transport. This needs **reactivity on validation** of code inspection due to the false positive in SAP Code Inspector that is handled by requesting an Exception. Let's see in detail what happened in development system:



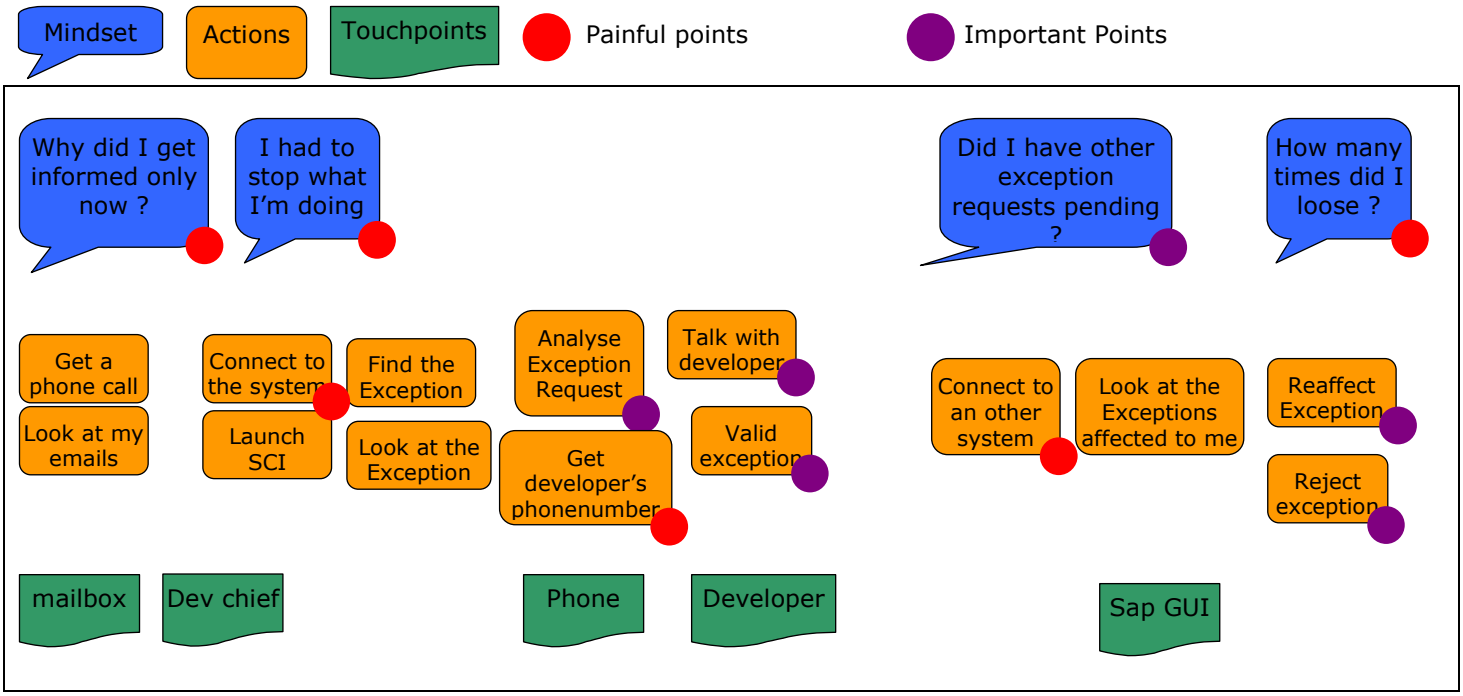
1. When developer has finished his work and want to release the transport request and Code Inspection check is done for the transport request's content.
2. If there is no error transport request is released 6.
3. If there is erros, developer could correct them 2 or ask for exception 3 (false positive).
4. This exception could be either rejected 4 or accepted 5 by QA and leads to transport request release 6

All of this must be very responsive:

1. QA team shouldn't loose time to identify the system (ECC, SRM, CRM, SCM, PI,…) where the exception request comes from.
2. nor loose time to connect to the system
3. email send at each new exception request.

4 The User experience Journey

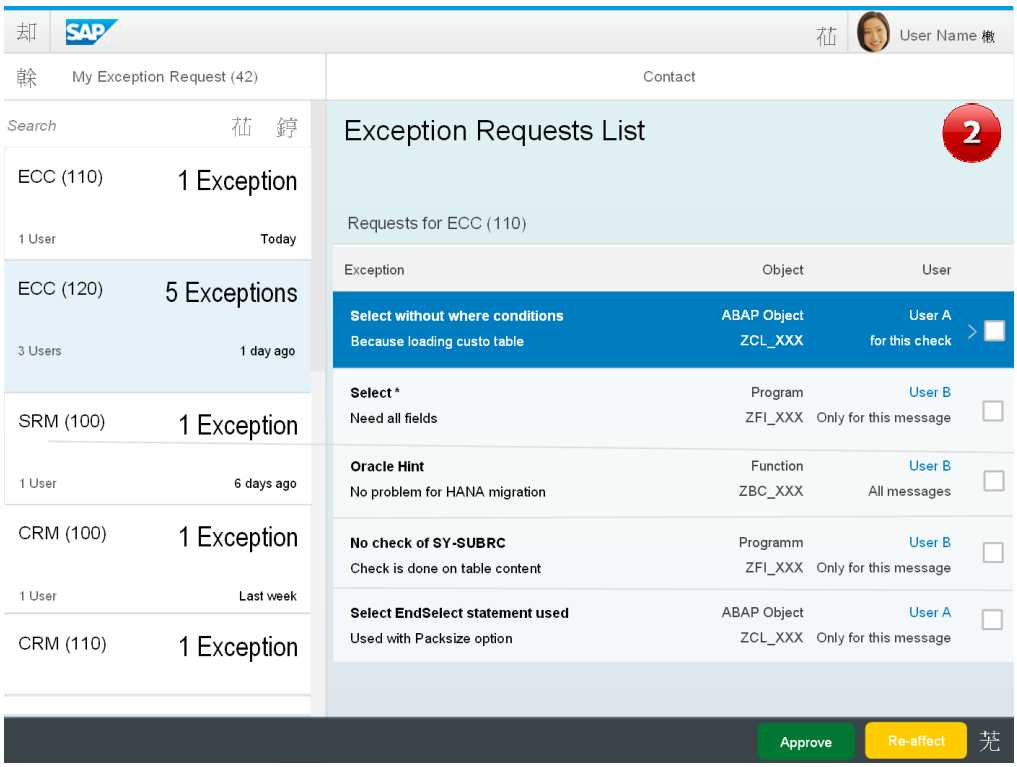
Like in Week 2 Unit 1, here are the legends of the user experience Journey for the "Wanted QA process":

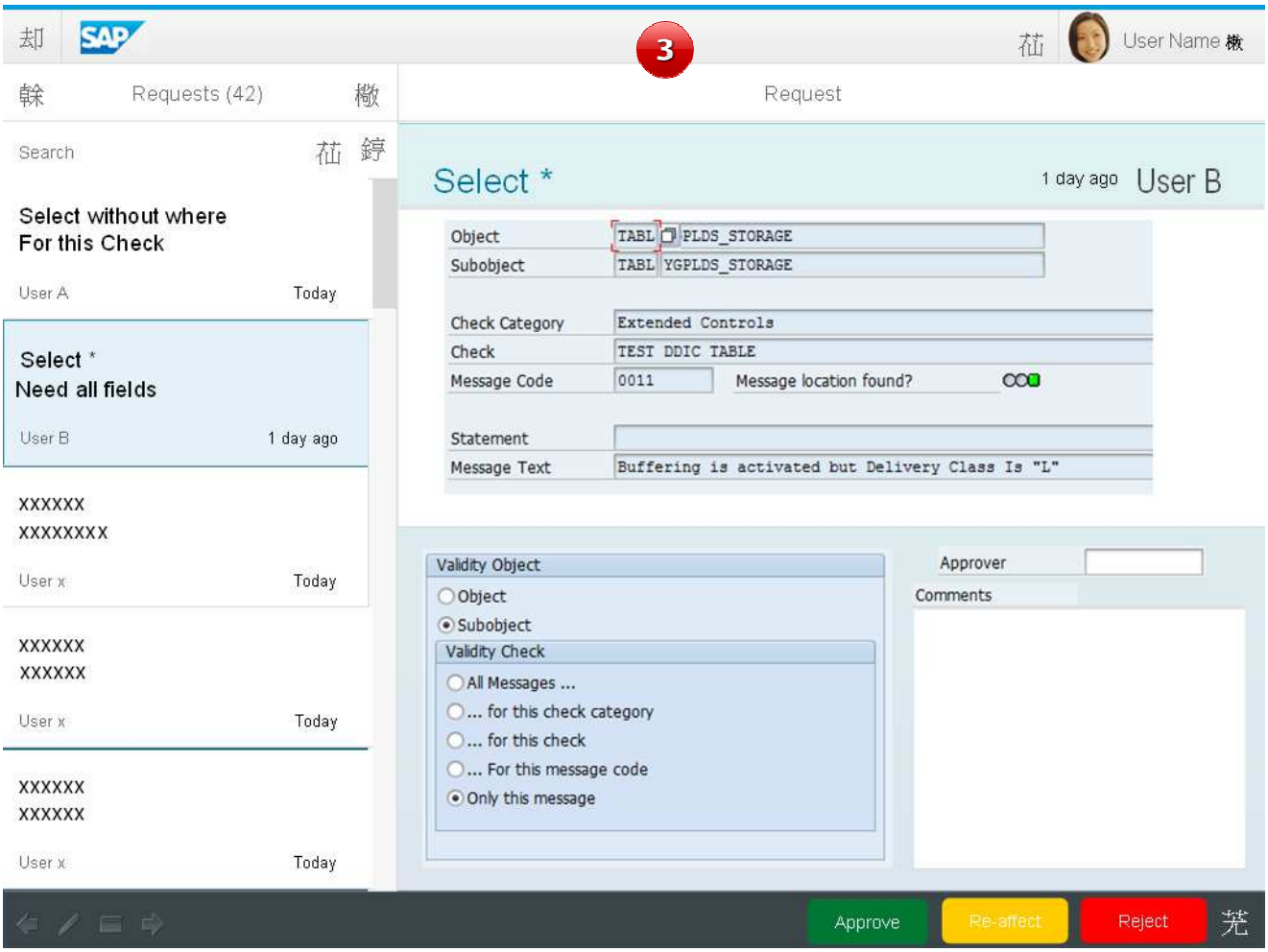


The most painful point is that QA validator is not informed when Exception request is set by developer, a lot a time is loose waiting for QA validation because QA is not aware he has something to validate. This often ends with a direct email or a phone call form Dev team to QA Validator. So notification flow is definitively a point of improvement. A second painful point is times taken to connect to systems and to find developer's phone number.

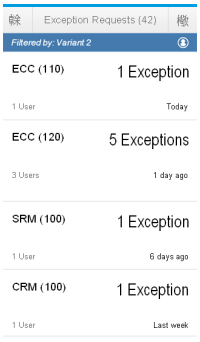
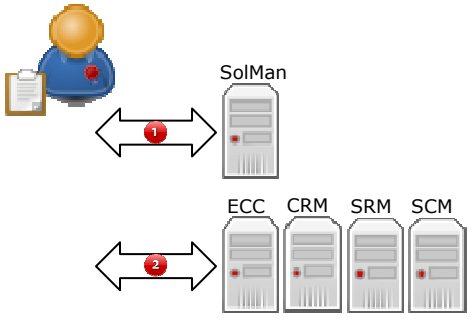
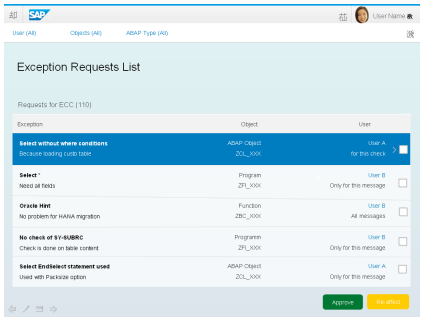
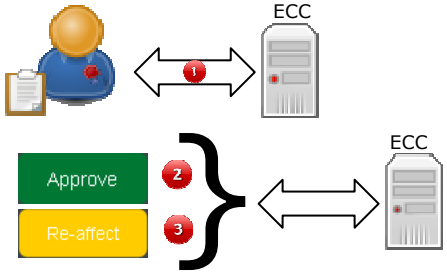
5 Mock UP

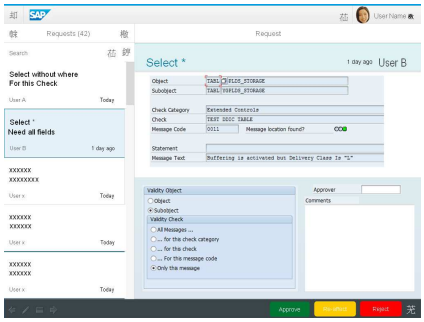
The template used is based on several templates: master/detail, full screen list and approval templates. The Screen #2 is a mix from Master/Detail where detail is a list with approval. The Screen #3 is the detail (from master/detail) with approval (or reject). The Fiori launch pad tile is presented as Screen #1. The followings screen have been created using protyping kit BUT without knowing all the limit and capacities of the UI component. Therefore the application screens might differs.

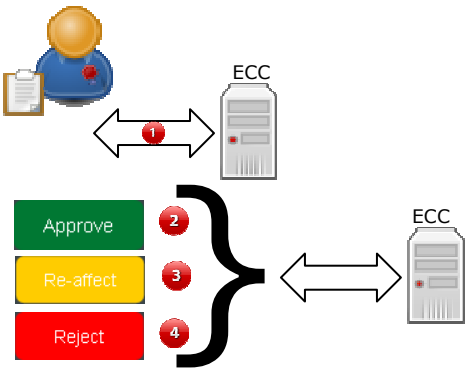




Here are the detailed steps (from 1 to 4):

Screen look & feel	Program logic	Actions description
	 Same as the next screen but just could total of requests	Tile: like many workflow approval indicate the total amount of request waiting for your approval or rejection. Should be 0 otherwise you've got urgent work to do.
		1 Get from Solution Manager the list os SAP system where my user do exists (can log on). 2 Connect to each of this system to get the list of exception that the connected user has to approve
		1 Get list of exception from ECC system that actual user has to approve. 2 The selected exceptions are approved, the developers are notified both by SAP message and email. 3 The selected exceptions are re-affected to an other user (QA team) for processing. This user is notified both by SAP message and email. He is now the approver of these exception requests

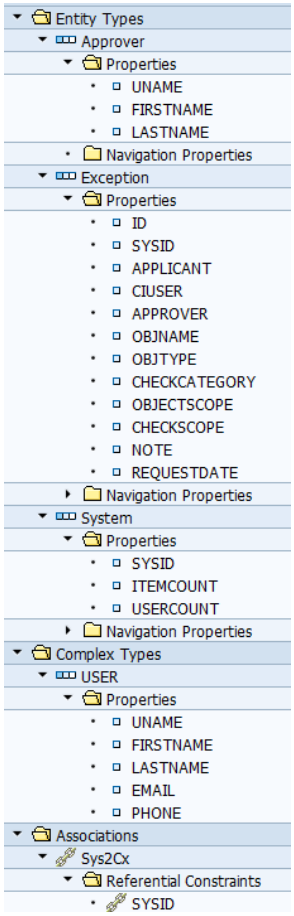




1. Get the detail of the selected exception
2. Exception is approved, the developer is notified both by SAP message and email.
3. Exception is re-affected to an other user (QA team) for processing. This user is notified both by SAP message and email. He is now the approver if this exception request
4. Exception is rejected with a comment, the developer is notified both by SAP message and email. Developper has to fix this SCI error or provide new justification for the exception request.

6 SAP Web IDE App

The Odata definition has been made on SAP NW Gateway using transaction SEGW :



The mockData have been created using json-generator here is the

```
[ '{{repeat(10)}}',
  { ID: '{{guid()}}',
    SYSID: '{{random("ECC (100)", "SRM (110)", "CRM (200)")}}',
    APPLICANT: { FIRSTNAME: '{{firstName()}}', LASTNAME: '{{surname()}}', EMAIL: '{{email()}}', PHONE: '+1 {{phone()}}' },
    OBJTYPE: '{{random("Class", "Program", "Table", "Function Group")}}',
    OBJNAME: function (tags) {switch(this.OBJTYPE) {
    case "Class": return 'ZCL_XXX';
    case "Program": return 'YPRG_XXX';
    case "Table":  return 'TXXX';
    case "Function Group": return 'YFG_BLABLA';          }          },
    CHECKCATEGORY: '{{random("General Checks", "Performance Checks", "Syntax Check/Generation")}}',
    REQUEST_TEXT: function (tags) {switch(this.SYSID) {
    case "ECC (100)":  return 'Today';
    case "SRM (110)":  return '3 days ago';
    default:  return '2 weeks ago';        }              },
    NOTE: '{{random("can\'t find any other solution", "Will fix later", "No an error for me", "See togheter on the phone")}}',
    CIUSER: { FIRSTNAME: '{{firstName()}}', LASTNAME: '{{surname()}}', EMAIL: '{{email()}}', PHONE: '+1 {{phone()}}' },
    OBJECTSCOPE_CODE: '{{integer(0, 1)}}',
    OBJECTSCOPE: function (tags) {switch(this.OBJECTSCOPE_CODE) {
    case 0: return 'Object';
    case 1:  return 'Subobject';          }              },    CHECKSCOPE_CODE: '{{integer(0, 1)}}',
    CHECKSCOPE: function (tags) {switch(this.OBJECTSCOPE_CODE) {
    case 0:  return 'All Messages';
    case 1: return 'for this check Category';
    case 2: return 'for this check';
    case 3:  return 'for this message code';
    default: return 'Only this message'; }              },
    Approver_code: '{{random("JDOW", "JSMITH", "PPERSON")}}',
    APPROVER: function (tags) { switch(this.Approver_code) {
    case "JDOW": return { "UNAME": "JDOW", "FIRSTNAME": "Jack", "LASTNAME": "Dow", "EMAIL": "Jack.Dow@sap.com", "PHONE": "+1 (193) 142-9142" };
    case "JSMITH": return { "UNAME": "JSMITH", "FIRSTNAME": "Jane", "LASTNAME": "Smith", "EMAIL": "Jane.Smith@sap.com", "PHONE": "+1 (193) 193-9191" };
    default: return { "UNAME": "PPERSON", "FIRSTNAME": "Paul", "LASTNAME": "Person", "EMAIL": "Paul.Person@sap.com", "PHONE": "+1 (193) 193-9142" };  }
    }          }              ]
```

Watch the demo on the video: https://youtu.be/ADYigLv6UUQ.