ELIXIR
INTELLIGENT SOFTWARE

بيئة الإكسير لتطوير البرمجيات
Elixir Framework

1 مقدمة

الإكسير بيئة framework تصريحية (Declarative) ذكية لتطوير وصيانة "الحلول البرمجية المؤسساتية الداعمة للوب"، بسرعة ونوعية جيدة وكلفة منخفضة.

في الواقع فإن النظم المؤسساتية مثل نظم ERPو CRM وBilling وBanking وE-government وHMIS وHIS وSIS وUMS وغيرها وصلت لدرجة من التعقيد لا يمكن للمهندس الجمع بين التعقيدات الوظيفية والتعقيدات الفنية مثل الأمن والأداء والاستقرار والتعقب والتنبيه ودعم الوب وتوزيع العبء والعنقدة وغيرها من المتطلبات غير الوظيفية. كما أصبح من الصعب صيانة هذه البرمجيات ونقلها من جيل من المهندسين إلى جيل آخر في الوقت الذي ازداد فيه الإقبال على هذه البرمجيات الإدارية (بل اصبحت أمرا لا غنى عنه من جهة) وازدادت كلفها من جهة أخرى.

يضاف للصعوبات السابقة الحاجة لتخصيص هذه البرمجيات من أجل كل مؤسسة على حدة مما يقسم البرمجية الواحدة برمجيات عديدة تحتاج لفريق ضخم لصيانتها مما يؤدي بها بسرعة إلى الشيخوخة عند أقل تبدل في أعضاء الفريق.

تتمثل شيخوخة البرمجيات بفقدان المطورين الذين ساهموا في وضع البنيان الأساسي لها وصعوبة تدريب مطورين جدد وفي حدوث تباين بين البرمجيات وبين وثائق التحليل وبين البرمجيات ووثائق التصميم وفي حدوث تغييرات في الرماز المصدر لبعض المكتبات التي دخلت في بنائه وكذلك انتقال لغات البرمجة ونظم التشغيل والأدوات المستخدمة إلى إصدارات أحدث ربما لا نستطيع معها إعادة ترجمة هذه البرمجيات بسهولة. ولا يبق سوى الرماز المصدر منطلقاً لفهمها وصيانتها.

في الواقع هنالك جانب وحيد من البرمجيات لا يشيخ بل يزداد شباباً مع الوقت وهو الخبرة في استعمال هذه البرمجيات من قبل المستخدم النهائي والذي يعرف مداخلها ومخارجها ونتيجة تنفيذ أي عملية. لذلك فكرت الإكسير بتطوير بيئة شفافة تمكن المطور من الانتقال مباشرة بين واجهات الاستعمال إلى الرماز الموافق لها بل زودته بلغة تصريحية لصيانة البرامج وتطويرها دون العودة للغات البرمجية التقليدية والمترجمات بل ودون إيقاف هذه البرمجيات.

.

الإكسير تقنية مبتكرة تدمج قدرة البنيان المشتق من النماذج MDA، وسهولة توصيف إجراءات العمل اعتماداً على القواعد rule based workflow ورحابة الوب،ومرونة البنيان الموجه بالخدمات AOP، وتوظفها في بناء نظم برمجية مؤسساتية داعمة للوب بأحدث المواصفات الفنية بزمن قياسي ونوعية عالية وكلف محدودة حيث أنها لا تتطلب خبرات طويلة لاستعمالها. ومن أهم ما تتميز به الإكسير هو سهولة المكاملة والصيانة وتطوير البرمجيات ونقل المشاريع البرمجية من مهندس لآخر ومن فريق لآخر. كما تسمح الإكسير بتطوير البرمجيات دون إيقافها أو إعادة ترجمة الرماز.

وقد مرت تكنولوجيا الإكسير بثلاثة مراحل في تاريخها:

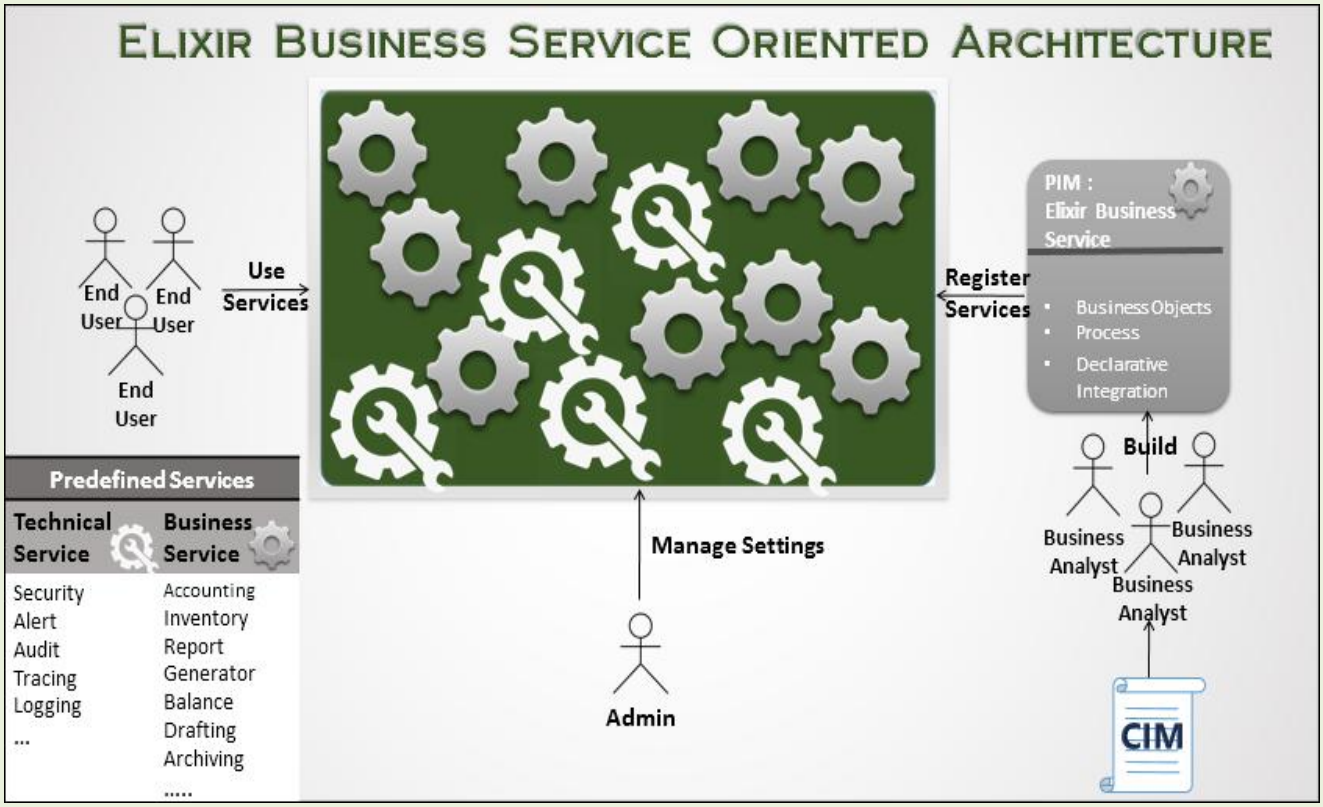
مرحلة وصِّف وانشر Deploy & Specify والتي تسمح ببناء البرمجيات بمرحلتين فقط وهي مرحلة توصيف البرمجيات ومن ثم نشرها مما **يوفر مرحلة التصميم والبرمجة**.

مرحلة شغل ثم وصّف Run then Specify والتي تسمح ببناء البرمجيات بمرحلة واحدة حيث يمكن تعديل التوصيف أثناء عمل البرنامج **وتوفير مرحلة النشر**

مرحلة **"WYSIWYP** ما تراه هو ما تبرمجه " What You See Is What You Program: حيث أصبح البرنامج يعكس الواقع الحقيقي ويمكن للمبرمج اللانتقال بين واجهات النظام والتوصيف بسلاسة مما يسمح لأي مبرمج بمتابعة البرامج التي قام بها لآخرين وأصبح شعارنا **" إذا كنت تستطيع استعماله فيمكنك برمجته وتطويره"**

يتمتع البرنامج الناتج بجميع الخصائص العصرية لبرامج الوب من أمن وظيفي، وتوزُّع على عدة محطات عمل، وتسامح في الأخطاء، وتجاوز للأعطال، وخدمات تنبيه وتنصت، وسهولة في الصيانة والتطوير، وتعدد لغات وتعدد قنوات الاتصال من وب، وواب، وبريد إلكتروني، وفاكس، ورسائل قصيرة، وغيرها.

2 وصف الإكسير



الإكسير بيئة **تطوير تفاعلية** ومتجانسة من الخدمات المتشابهة شكلاً وإن اختلفت بدلالاتها فمنها ما هو فني Technical مثل الأمن والتعقب والتنصت وتوليد التقارير ومنها ما هو وظيفي Business مثل المحاسبة والمستودعات والأرشفة وإدارة المسودات وإدارة الأرصدة واي مكونات وظيفية أخرى.

للإكسير ثلاثة أنواع من المستخدمين ولكل نوع عدد غير محدود من المستخدمين الفعليين القادرين على العمل على التوازي وهم محلل النظم ومدير النظام والمستخدم النهائي:

2.1 محلل النظم

يقوم **محلل النظم** Business Analyst في أي وقت بإضافة خدمات للمنظومة. حيث يبدأ عمله بوضع التحليل النصي CIM الذي يوصف المسألة والمصادقة عليه مع الزبون ومن ثم تحويله إلى تحليل صوري PIM وإدخاله للإكسير. وتتكون الخدمة Service في الإكسير من ثلاثة مكونات تصريحية Declarative

1- نموذج الأغراض Business Objects
2- نموذج الإجراءات Process
3- نموذج التكامل مع بقية الخدمات المعرفة مسبقاً في النظام Declarative Integration.

أي كما هو موضح في المخطط أعلاه أن المحلل عليه فقط أن يوصف المشكلة ويحولها إلى نموذج الخدمة في لغة الإكسير لكي تقوم بدورها ببقية مراحل العمل من تصميم وتحقيق آلياً.

2.2 مدير النظام Admin

والذي يقوم بضبط إعدادات النظام والخدمات المتوفرة مثل إضافة مستخدمين وتعديل صلاحياتهم وضبط إعدادات التنصت والتعقب وغيرها من الخدمات

2.3 المستخدم النهائيEnd User

ويُقسم وفقا للتحليل الذي قام به محلل النظم إلى أنواع لكل منها صلاحياته فمنهم من يستعمل الخدمات الوظيفية ومنهم من يحق له ضبطها ومنهم من يحرر ومنهم من يصادق ومنهم من له حق التعديل والإلغاء

2.4 Elixir CIM

CIM أو Computational Independent Model

لغة محكية لتوصيف المسألة من وجهة نظر المستخدم. هذه اللغة وإن كانت محكية فإن العبارات فيها تُكتب وفقا لاستمارات تسهل الانتقل منها إلى النموذج الصوري دون أن تعيق قدرة الانسان العادي (المستفيد) على قراءتها بل أنها أسهل من اللغة الحرة في كونها تحفذ عند القارئ توقع الجملة التي يقرؤها. تتضمن Elixir CIM ما يلي:

- **مقدمة تحوي وصف للمشكلة واقتراح حلول لها.**
- **الخدمات التي يقدمها النظام .**
- **سير عمل هذه الخدمات بحيث يتضمن سيناريو سير عمل كل خدمة مايلي :**

من يقوم (مستخدم أو دور أو منصب) و**كيف يصل** (مدخل في القائمة الرئيسية أوعمل تابع لغرض أو عمل مستقل) و**ماذا يستطيع أن يفعل** (عمليات وظيفية أو مهمات) و **ماذا يرى** (نمط الأغراض المطبقة عليها العملية وشروطها وخصائصها)

مثال: يقوم مسؤول الحجوزات **(من)** باختيار المدخل "حجوزات" من القائمة الرئيسية **(كيف يصل)** ويرى جميع الحجوزات الغير منتهية **(ماذا يرى)** ويستطيع إنشاء حجز جديد أو إلغاء حجز سابق أو الاطلاع على تفاصيل حجز قديم أو تأكيد حجز **(ماذا يستطيع أن يفعل)**

- **أسماء الأغراض المطبقة عليها هذه الخدمات والموجودة في النظام وحقوق استعراض هذه الأغراض.**

ملاحظة :نستخدم نماذج جمل تحوي كلمات مفتاحية تسمح للمستخدم بالانتقال من اللغة المحكية بسهولة إلى النموذج الصوري PIM للغة نمذجة إلكسير أي كلما كان التحليل قريباً من بنية لغة الإكسير للنمذجة كلما كان تحويله وإدخاله للنظام أسهل.

2.5 لغة الإكسير لنمذجة الخدمة Elixir PIM

الـPIM هي لغة مستقلة عن منصة العمل وترمز لـ Platform Independent Model .

ويتألف Elixir PIM من ثلاث مكونات تحليلية وهي توصيف الأغراض الوظيفية Data Object وتوصيف الإجراءات Processes وتوصيف التكامل Integration والتي سيرد شرحها لاحقاً.

2.5.1 Data Object Specification (Class Model, ADT, etc.)

توصف الأغراض الوظيفية من صفوف وحقول وعلاقات بين الصفوف في الخدمة

1. Class description
2. Field descriptions
3. Relation descriptions

من الأمثلة عن بنى المعطيات: توصيف الأشخاص والشركات واستمارات العقود وعمليات البيع وغيرها

تتضمن لغة الإكسير لنمذجة الأغراض معطيات سامية أغنى من اللغات متعددة الأغراض General Purpose Languages نذكر منها:

1. For Class
   a. Data source ID
   b. Label
   c. Integrity
   d. Code
   e. Unique groups
   f. Indices
   g. Transient or persistent
2. For fields and relations
   a. Is mandatory
   b. Is final
   c. Is unique
   d. Default value
   e. Calculation formula
   f. DB format
   g. Pattern
   h. Possible values
3. For relations
   a. Foreign keys/primary keys
   b. Multiplicity min-max
   c. Symmetric name

اعتماداً على هذه المعلومات السامية يتمكن محرك الإكسير وجون أن تدخل من المستخدم من بناء قواعد البيانات والاستعلامات المرتبطة بها وكذلك بناء واجهات الاستخدام ومحركات التفاعل معها.

2.5.2 Process Specification

توصف إجراءات العمل وهي سلسلة من الأحداث مثل المهام الآلية والمهام اليدوية واستيقاظ مؤقت أو وصول رسالة وتهدف هذه الأحداث معاً إلى إنجاز مهمة متكاملة من الخدمة.

من الأمثلة عن الإجراءات: مراحل عمليات البيع والشراء والأصول الثابتة والمحاسبة والتحويل الصيرفة، إلخ.

أيضاً تقدم الإكسير لغة مفسرة خطاطية scripting language تسمح بربط الإجراءات بنموذج الأغراض السابق بعبارات بسيطة وقصيرة جداً من الامثلة عليها مداخل القائمة الرئيسية والمهام اليدوية والمهام الآلية.

2.5.2.1 *تعريف مهمة يدوية*

الشكل العام لتعريف مهمة يدوية هو كما يلي

```
Apply <actions> on <target> show <keys> where <condition>
```

حيث تدفع العبارة السابقة محرك الإكسير إلى إظهار صفحة HTML تتضمن مجموعة من الأعمال <actions> لتنفيذها على غرض أو مجموعة من الأغراض <target> والتي تحقق الشرط <condition> وتسمح للمستخدم برؤية <keys>. وفميا يلي مثال

```
Apply[save, validate, cancel] on Sales show [id, counterpart, transactions]
                                where status==pending && client.category=1
```

كما يبين المثال فإن لغة الإكسير ثلاثية الأرتال فالتعليمة السابقة سينشأ عنها صفحة HTML تتضمن إمكانية استعمال العمليات save, validate and cancel وستظهر الحقول id, counterpart and transactions ولكن ليس لجميع الأغراض لأنها ستبحث ضمن قاعدة البيانات آلياً عن الأغراض التي تحقق الشرط status==pending &&

client.category=1. كما نلاحظ أن الشرط السابق يتضمن leftJoin ضمني بين البيع والزبون وبين الزبون وشريحته category وهي عملية تحتاج لأسطر في لغة SQL للتعبير عنها أما الإكسير فقط استفادت من المعلومات السامية عن الغرض لتوليدها آلياً.

في الواقع التعليمة السابقة تكافئ عشرات الأسطر من SQL إضافة لمئات الأسطر من الـHTML ومئات أخرى من Business Logic server.

2.5.2.2 تعريف مهمة آلية

الشكل العام لتعريف مهمة آلية هو كما يلي

```
Apply <expression> on <target> when <target> is (created, updated, or deleted) while satisfying <condition>
```

بعبارة أخرى فإن النظام سينفذ التعبير <expression> عندما يقوم أحدهم بإنشاء أو تعديل أو حذف غرض من نوع <target> شرط أن يحقق الشرط <condition>.

مثال

Apply generate payment on Sale when Sale is updated while status==validated.

كما تتضمن اللغة أنواع أخرى من الخطوات في الإجراءات منها المؤقتات الزمنية والأعمال الخلفية.

Declarative Integration 2.5.3

المكاملة مع الخدمات المعرفة مسبقاً Predefined Services التقنية منها technical أو الوظيفيةBusiness وتوصيف إعداداتها تصريحياً باستعمال لغة الـTDL السابقة وتحديداً المهام الآلية:

- Accounting Impact آثار العمل المحاسبية (المالية).
- Inventory Impact آثار العمل المستودعية.

من الأمثلة عن الآثار المحاسبية: يترتب ذمة على الزبون مباشرة في حال إبرام عقد بيع في حين تترتب ذمة له في حال إبرام عقد شراء واستلام بضاعة.

من الأمثلة عن الآثار المستودعية: ترتفع قيمة مخزون الشركة في حال إبرام عقود شراء مباشرة وتنخفض قيمة المخزون في حال عقود البيع ويرتفع قيمة المحجوز بآن واحد.

- Report Generator إعدادات التقارير المطلوبة الخاصة بالخدمة باستخدام المجمع (مولد التقارير)

من الأمثلة عن التقارير: تقرير أيام عدم البيع لكل مندوب مبيعات.

- إعدادات الأمن Business Security

توصيف للنموذج الأمني المتبع في الإكسير بحيث يعتمد على بعدين وهما:

1. أمن العمليات (وفيها تعريف صلاحيات العمليات للأدوار)
2. أمن المعلومات (حقوق قراءة المعلومات الخاصة بالأدوار حسب الصلاحيات)

من الأمثلة عن أمن العمليات: مسؤول بريد صادر يستطيع طباعة بريد صادر والاطلاع على البريد المعلق.

من الأمثلة عن أمن المعلومات: مسؤول البريد الصادر لا يستطيع الاطلاع إلا على البريد الخاص بالمجموعة التي يعمل ضمنها.

- وغيرها من الخدمات الوظيفية من إعدادات الموازنة والمسودات والمهام والديوان...
- وغيرها من الخدمات التقنية من إعدادات التنصت والمتابعة ...

3 نقاط تمييز الإكسير USP (Unique Strength Points)

توجد العديد من بيئات العمل التي تسمح بتطوير برمجيات داعمة للوب مثل Ruby on rails, Struts, or Spring إلا أن الإكسير تتميز عنها بنقاط هامة جداً وهي:

1- تشاركية **Cooperative**: يمكن لأكثر من مطور العمل معاً على المنظومة واختبارها. كما أن أداة الإكسير للتنقيح (Elixir Debugger) تسمح لكل مطور العمل منفردا دون التأثير على الآخرين لانه محصور بنياسبه (threads) فقط .

2- تصريحية **Declarative**: لا تتضمن آلية التطوير في الإكسير أي Control Structure بل هي طريقة تصريحية تسمح لمحلل النظم بتطوير المنظومة من خلال ملئ عدد قليل من الاستمارات مسبقة التعريف.

3- تفاعلية **Interactive**: يمكن الانتقال بسهولة وبخطوة واحدة بين الواجهات والتصريحات المقابلة

4- **التطوير أثناء التشغيل Run Then Specify** : لا تحتاج تطويرها إلى إيقاف البرمجية ولا إلى إعادة نشر فقط تسجيل الخطوة مما يوفر زمن طويل في الترجمة وإعادة النشر.

5- **محكمة coherent**: تقم الإكسير آلياً بعكس أي تعديل أو تطوير على الطبقات الثلاثة من client, business, storage دون تدخل من المستخدم، فعند إضافة حقل جديد إلى غرض أو خطوة جديدة إلى إجراء ستقوم الإكسير آلياً بتوليد انعكاساته على واجهة المستخدم وقاعدة البيانات والخدمات في آن واحد.

6- **سهلة الاستعمال والصيانة**

a. **شغل ثم وصف Run Then Specify**: لا يوجد مراحل طويلة في التطوير مثل التصميم والبرمجة والترجمة والنشر بل يوجد مرحلة واحدة وهي التوصيف فمجرد إدخال التوصيف يمكن استعمال البرنامج.

b. **ما تراه هو ما تبرمجه WYSIWYP**: لا يوجد فصل بين "التوصيف الصوري" و"البرنامج" حيث يمكن للمبرمج أثناء استعماله للمنظومة أن ينتقل فورا من الصفحة التي يعمل عليها إلى النموذج الصوري المناسب بطريقة تفاعلية ويصل بسهولة إلى ترجمة الصفحة، رسائل الخطا ورسائل التحذير وتوصيف أعمال الصفحة وتوصيف المعلومات التي تظهر في الصفحة والشروط والقيود وغيرها.

c. **لا تحتاج لفهم البنية التحتية (إذا كنت تستطيع استعماله فيمكنك برمجته وتطويره)**: كل ما نحتاجه هو فهم جيد للمسائل الوظيفية ولاحاجة لخبرة في مجال الـHTML أو JavaScript أو Data Base أو Enterprise Components أو Application Server أو Web servers.

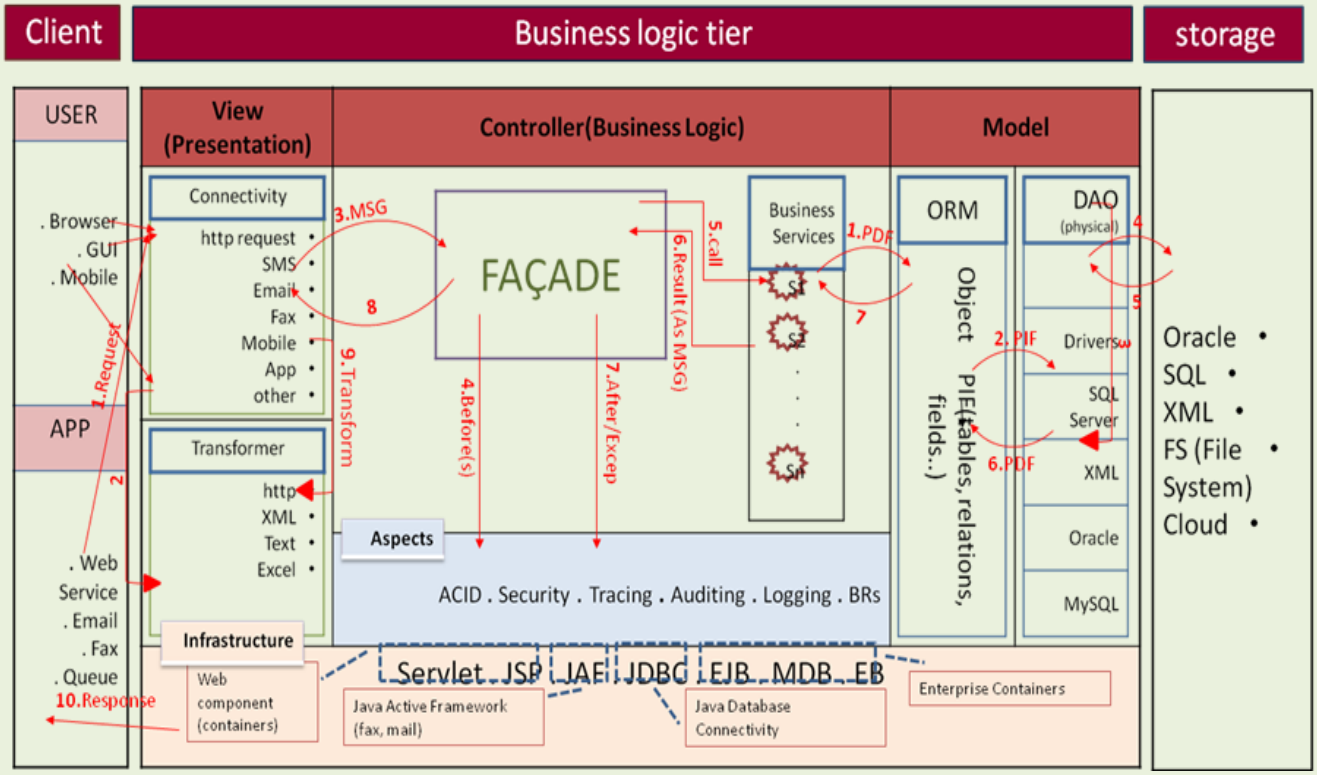
d. **لا نحتاج لتعلم برمجة Zero code programming:** بناء منظومة يتم بملئ استمارات أي حقول وأعقد الحقول هي تعبير منطقي وهذه التعابير مكتوبة بدلالة المسميات الوظيفية ولا حاجة لمعرفة بقواعد البيانات أو بتطوير واجهات الوب.

7- **سهولة في إدارة المشروع manageability:** في الواقع فإن برامج الإكسير البرنامج قابل للعد **Quantifiable** والمتابعة فالبرنامج هو مجموعة من الخدمات تتألف كل منها من مجموعة من الاستمارات ومجموعة من الخطوات. الخطوة والاستمارة تتكون من مجموعة من الحقول وعليه فمن الممكن قياس مقدار التقدم الفعلي على عكس البرمجة التقليدية.

8- **سهولة تبادل المشاريع بين المهندسين:** يكفي أن يفهم المهندس كيف يستعمال البرنامج ليتمكن من صيانته وتطويره وهذا يعالج مشاكل التبدل الكبير في فريق العمل.

4 بنيان الإكسير Elixir Architecture

على الرغم من أن الإكسير توفر على المطور التفكير في القضايا غير الوظيفية إلا أنها مبنية على بنيان مؤسساتي Enterprise Architecture يدعم جميع المتطلبات غير الوظيفية العصرية والضرورية للتطبيقات المؤسساتية



4.1 دعم لنماذج التصميم العصري

. MVC, AOP, MDA, Façade, Lazy List, 3Tiers, ORM, DAO ومنها

4.2 داعمة للويب Web Based

يمكن الدخول إلى النظام من أي حاسب يمتلك متصفح إنترنت ويستطيع الوصول للمخدم.

يمكن تقييد الدخول إلى المخدم من الشبكة المحلية فقط LAN أو الإنترانت (intranet) أو السماح بالوصول إليه من أي مكان في العالم (internet).

يجري التحقق من هوية المستخدم باستعمال كلمة سر ويمكن أيضاً اللجوء لكلمتي سر أو أكثر يمتلك كل شخص كلمة منها، وذلك للعمليات الحرجة والحساسة، اعتماداً على محرك الإكسير لإجراءات العمل، كما تتضمن آليات للتحقق من أن المستخدم بشر وليس آلة.

4.3 تسامح مع الأعطال

يستطيع النظام التعامل مع العديد من أنواع الأعطال وتجاوزها مثل أعطال البرمجيات والعتاديات وقواعد المعطيات والشبكة، وذلك من خلال:

- ✓ آليات العنقدة **Clustering** حيث يمكن إرساء النظام البرمجي على عدة مخدمات سواء كانت متواجدة في نفس الموقع أو في مواقع متفرقة، ويستمر النظام بالعمل في حال توقف أحدها.
- ✓ آليات الـ **Replication** حيث يمكن إرساء قاعدة المعطيات على أكثر من مخدم، وتبقى قاعدة المعطيات متاحة في حال توقف أحد هذه المخدمات.
- ✓ الاستقلال عن كيفية **الربط الشبكي** حيث يمكن الوصول للنظام من طرائق عدة LAN, Intranet, Internet ويستهلك القليل من موارد الشبكة مما يسمح باستعماله حتى عن طريق Dial-up وبالطبع عن طريق الدارات المؤجرة وDSL والهاتف الجوال.

4.4 مزودة بمحرك لإدارة تدفق إجراءات العمل Workflow Engine

يتضمن النظام محركاً لإجراءات العمل أو Workflow Engine. وبذلك يؤمن تقسيم العمل بين الأدوار المختلفة، بحيث لا يرى المستخدم إلا الجزء المتعلق بدوره مما يخفف من التعقيد الظاهري للنظام.

يدعم محرك النظام نوعين من تدفقات إجراءات العمل: النوع المعتمد على مخططات تدفق مثل BPMN والنوع المعتمد على قواعد العمل Rule Based Engine.

يعتمد النظام مبدأ تخاطب مبسط مشابه لنظم البريد الإلكتروني، حيث تصل المهام الخاصة بموظف معين إلى ما يشبه بريده الإلكتروني. وعند تنفيذه لمهمة ما فإن النظام يقوم بتحويلها آليا إلى الموظف المسؤول عن المرحلة التالية بناءً على تسلسل إجراءات العمل التي جرى تفصيلها في مرحلة التحليل.

يستطيع مستخدم ذو صلاحيات خاصة إعادة تحويل المهام يدوياً من موظف إلى آخر. تفيد هذه الخاصة في ضمان استمرار العمل في حالات الغياب الطارئ لأحد الموظفين.

يستطيع مستخدم ذو صلاحيات خاصة أن يحدد آليات توزيع العبء بين الموظفين الذين يمتلكون أدواراً متشابهة، كأن يهتم بعضهم بزبائن مدينة معينة أو حي ويهتم الآخر بمدينة أو حي آخر.

4.5 أمن وظيفي متقدم

- ✓ يعتمد النظام مبدأ الأمن الوظيفي المعتمد على الأدوار الوظيفية، مثل دور مدير، ودور معاون مالي، ودور معاون إداري، إلخ.
- ✓ تنحصر مهمة مدير الموارد البشرية في إضافة موظفين جدد وإعطائهم أدواراً تتوافق مع قرارات تعيينهم وسيحصول آلياً على الصلاحيات الموافقة لمناصبهم.
- ✓ يمكن لمدير ذو سلطة أعلى إعادة توزيع الصلاحيات بين المناصب الوظيفية المختلفة إذا اقتضت الضرورة ويمكنه التحقق من عدم وجود اختراقات أمنية مفتعلة، من خلال تقرير يظهر المستخدمين المعرفين على النظام والأدوار المسندة إليهم.

- ✓ يستطيع النظام أن يعرّف سماحيات تتعدى الحدود الاعتيادية التي تعتمد على قواعد المعطيات، حيث يمكن تعريف قيود أمنية على مستوى **الغرض**، أو على مستوى **حقل** معين، كأن نسمح لمدير باستعراض ملفات الموظفين التابعين له ونخفي فقط الحقل المتعلق برواتبهم. أو تعريف

سماحيات مرتبطة **بالزمن**، كأن نتيح لمدير ما رفض طلبات معينة وذلك قبل قبولها حصراً. وأخيرا يمكن ربط إمكانية الدخول للمنظومة بعناوين **IP** خاصة بكل مستخدم على حدة.

- ✓ يقوم محرك الإكسير للأمن الوظيفي **بالتنصت** وتدوين جميع الحركات اليومية مما يسمح بمعرفة من قام بأي عمل ومتى ولماذا.
- ✓ يقوم محرك الإكسير أيضاً بتسجيل أوقات دخول وخروج المستخدمين ويعطي تقريراً بذلك للمستخدم المسؤول.
- ✓ يستطيع محرك الإكسير للأمن مراقبة جميع مداخل النظام وليس فقط واجهة الاستخدام.

4.6 مستقل عن منصة العمل Platform independent

يتميز النظام كونه مستقل عن منصة العمل من حيث أنظمة التشغيل ومخدمات التطبيقيات وقواعد المعطيات حيث يمكن للنظام أن يعمل على:

نظام التشغيل: Windows أو Linux
مخدم تطبيقات Jboss
قاعدة معطيات MySQL أو Oracle أو Postgres أو Sql server

4.7 سهولة التطوير والصيانة

لغة الإكسير هي لغة تصريحية Declarative Language تجمع بين مميزات الـ Lisp في البرمجية التابعية وProlog في البرمجة المنطقية وJavaScript في البرمجية الديناميكية كونها تسمح بإضافة حقول وطرائق أثناء التشغيل Runtime ولا تحتاج برامج الإكسير في تطويرها إلى إعادة ترجمتها Compiling ولا إلى إعادة نشرها. في الواقع فإن رمازها المصدر هو لغة صورية مقروءة ومحمولة على البرنامج نفسه يمكن للمطور متابتها وتعديلها من خلال **محرر توصيف صوري** سهل الاستعمال يسمح بـ

1- إضافة تقارير جديدة
2- إضافة واجهات تفاعلية جديدة
3- إضافة مداخل إلى القائمة الرئيسية
4- تعريف توابع جيدة
5- تعريف مهام جديدة
6- تعريف آثار وظيفية جديدة كأن نعرف أثر عمل وظيفي على المحاسبة والمستودعات والمالية
7- إضافة إعمال خلفية بهدف التنصت أو لأهداف وظيفية أخرى
8- إضافة نماذج جديدة من صفوف وأغراض وربطها مع قواعد البيانات

جميع هذه المهام يمكن القيام بها دون إيقاف البرمجية عن العمل وبشفافية مطلقة حيث لا يحتاج المطور لمعرفة أي معلومات عن قواعد البيانات والـhtml والمكونات المؤسساتية الأخرى.

5 التحديات التي عالجتها الإكسير

طُورت الإكسير لمواجهة مجموعة من التحديات التي تواجه عملية تطوير البرمجيات المؤسساتية والتي تتزايد في ظروف تتبدل فيها فرق التطوير بمعدل عال:

5.1 مكاملة سهلة لعدد كبير من المكونات

في الواقع فإن المشكلة في الترابط بين المكونات البرمجية ليست فنية وإنما وظيفية ولذلك فلا بد، لمن يطور مجتزءاً جديداً، من فهم كامل لمنطق عمل المجتزآت العمودية ليرسل المعلومات الصحيحة في اللحظة المناسبة، بل ويكون قادراً على التراجع عنها وتصحيحها، وأن يكون قادراً على مزامنتها كأن يزامن طلب الشراء مع حجز المبلغ في الموازنة، وأن يزامن استلام البضاعة من المورد مع تحقيق الدين الفعلي، وان يزامن قبول الفاتورة من المزود مع الفحص الفني للبضاعة، وأن يزامن إعطاء أمر الدفع مع نقل الاستحقاق المالي للمورد، وأن يزامن توزيع نفقات السيارة على المشاريع مع إغلاق السنة المالية، وأن يجعل نظام الأصول الثابتة يرى هذا المنتج الجديد ذو الخصائص الجديدة على أنه أصل ثابت. في الواقع ما نحتاجه في التجميع ليس فنياً بل ليس مهندساً حتى، بل مهندس بخبرة وظيفية قادر على فهم مثل هذه الحيثيات ولذلك نرى أن أوراكل وساب تقدمان أثناء التحقيق خبراء متفرغين بأسعار غالية جداً للقيام بعملية التحقيق ولذلك نجد أن البرمجيات المؤسساتية أصبحت من الاختصاصات المشتركة بين المعلوماتية وإدارة الأعمال ورأينا ظهور اختصاصات جديدة مثل MIS (نظم إدارة المعلومات) تدعي أنها تغطي الحلقة المفقودة بين المعلوماتية والإدارة.

عالجت الإكسير قضية المكاملة من خلال مكونات تكامل تصريحية declarative تسمح لطرف ثالث بالتنصت على أي مجتزأ أو صف أو غرض وإرسال هذا الحدث إلى آخر أو صف أو غرض آخر دون التعديل على كلا الطرفين.

من هذه المكونات التصريحية نذكر:

1- الدامج **Merger** والذي يستطيع القيام بدمج افتراضي لغرضين أو أكثر طورها أشخاص مختلفون وكأنهما غرض واحد ويتضمن هذا الدمج دمج افتراضي لجدوليهما في قاعدة البيانات وإن لم يكن هنالك علاقة مسبقة بين الجدولين

2- **الآثار المحاسبية والمستودعية**: تطبيق نموذج التصميم **IoC/AOP** على المكونات الوظيفية مثل المحاسبة والمستودعات والصرافة والموازنة وغيرها بحيث نتمكن من الاستفادة منها في الإجراءات المختلفة دون استدعائها صراحة ودون أن تستدعينا هي صراحة بل من خلال إعدادات وسيطة

3- **ادوات التهجير التزايدية** التي يمكن إجراؤها على مراحل حتى بعد البدء بالاستعمال الفعلي للبرمجيات

4- أدوات **تنظيف قواعد البيانات** من المعلومات الخاطئة التي قد تدخل عليها في بدايات استعمالها وخاصة أثناء استيراد المعلومات القديمة وارتكاب أخطاء في الاستيراد واكتشاف الخطأ في مراحل متأخرة بعد استعمالها تسمح هذه الأدوات بالتراجع آلياً عن مسارات محددة من الأخطاء بدلا من العودة إلى نسخة إحتياطية قديمة من قاعدة البيانات تتسبب ليس بالتراجع عن المعلومات الخاطئة فحسب بل أيضاً عن المعلومات الصحيحة. من الأمثلة استيراد سيارة بمعلومات خاطئة من حيث تاريخ شرائها وسعرها ومن ثم قيام أحدهم بعملية استهلاك تشمل هذه السيارة ونقل معلومات المالية إلى مجتزأ المحاسبة ووقيام مجتزأ التعقب والمراقبة بتحديث سجلاته بذلك وبعد كل هذه الأعمال التي تمت خلال دقائق نكتشف بأن هنالك خطأ في معلومات هذه السيارة ونريد التراجع عنه دون العودة إلى نسخة قديمة من قواعد البيانات ودون تجنيد عدة مهندسين لتحليل قاعدة البيانات والبحث في العمليات التي تمت منذ استيراد السيارة وحذفها وضمان صحة وتكامل قاعدة البيانات بعد الحذف.

5- **نقاط المراقبة الديناميكية** والتي تسمح باستخراج تقارير دعم قرار تجميعية غنية جداً ومترابطة ومتزامنة بين المجتزآت المختلفة بصرف النظر عن هيكلية كل مجتزأ على حدة. على سبيل المثال يمكن استخراج تقارير مالية دقيقة جداً عن المبيعات وفقا للمواد والأصناف والزبائن والدول والمدن وشرائح الزبائن بصرف النظر عن بنية الشجرة المالية. هذا سيجعل التعديلات على أي مجتزأ تبقى ضمن المجتزأ ولا تتعداه لمجتزآت أخرى كما يحدث عندما نضطر لتغيير في بنية الشجرة المالية المحاسبية لأننا نريد استخراج تقارير عن المبيعات أو المشتريات أو الموارد البشرية وما ينجم عن ذلك من صعوبات.

5.2 تخفيض كلف التطوير والصيانة

من السهل أن تدفع مؤسسة أو فرد مبلغاً كبيراً على إصلاح عطل في سيارة أو في منزل لفني بسيط في حين لا تقبل دفع نفس المبلغ بنفس ساعات العمل لمهندس معلوماتية قام بتعديل أو تطوير والسبب هو غياب البعد الفيزيائي للخدمة التي قدمها. وللوصول لكلف صيانة وتطوير منطقية قامت الإكسير بما يلي:

1- توفير زمن الترجمة والنشر واعتماد مبدا **Run then Specify** للتخفيف من زمن وكلفة التعديلات الصغيرة وإضافة التعديلات دون إيقاف البرنامج وبدون عملية ترجمة. وبذلك استطعنا الانتقال من حوالي 16-56 مهندس/ ساعة للتعديل إلى 2-8 مهندس/ساعة للتعديل

2- دمج الرماز المصدر والرماز التنفيذي في مفهوم واحد هو **"التوصيف"**. في الواقع فإن برامج الإكسير أصبحت تتضمن بطريقة ما الرماز المصدر لها ووثائق المساعدة وأدوات التطوير مما يخفف من مخاطر ضياع جزء من الرماز المصدر ويغني عن عمليات إدارة الإصدارات.

3- **البرمجة التفاعلية WYSIWYP**: امكانية الانتقال مباشرة من واجهات الاستعمال إلى التوصيف وبالعكس مما يوفر على المطور الزمن اللازم للبحث عن الرماز الموصف لواجهة أو تابع أو مهمة أوغيرها من المكونات.
4- **الهندسة العكسية الآلية Reverse Engineering**: والتي تسمح باستعراض مخططات UML للصفوف والأغراض الموجودة ضمن البرنامج.

5.3 تأمين اليد العاملة الخبيرة وتدويرها

عادة عندما تحتاج الصناعات الأخرى لمهندسين خبراء فإنهم يجدونهم فأي مهندس مدني خبير يمكنه متابعة مشروع بناء اعتماداً على المخططات. كما يمكن لطبيب خبير المباشرة في أي مشفى والقيام بعمله على أكمل وجه من اليوم الأول. في الواقع فإن لمخططات الأبنية معاييرها والإنسان هو الإنسان حيثما كان. أما البرمجيات فتختلف اختلافاً كبيراً فيما بينها مما يعقد عمليات استلامها وتسليمها بين المهندسين. في الواقع فإن خبرة مهندس المعلوماتية لا تسمح له باستلام مشروع من مهندس آخر بزمن أقل من شهر وغالباً ما يترك نقل مشروع أو عمل برمجي من مهندس لآخر آثراً سيئاً بشكل شبه دائم. هذا طبعاً إن كان جميع العاملين في مجال تطوير البرمجيات هم من المختصيين فهندسة البرمجيات ليست محمية بأي قوانين كما هو الحال في مجالات أخرى مثل الهندسة المدنية والميكانيكية والطب.

لذلك قامت الإكسير باعتماد مبدأ WYSIWYP لتسهيل الانتقال والصيانة كما ورد في فقرة الصيانة والتطوير.

5.4 التوثيق

ربما يعتقد البعض بأن التوثيق هو حل لمشكلة تدهور خبرة الفريق مع التغييرات المتتابعة في أعضائه. ولكنه، وللأسف، في الوقت الذي يعتمد مهندسي المدني والميكانيك على هذه الوثائق والمخططات بشكل أساسي في عمليات الاستلام والتسليم فإن الوثائق في المشاريع البرمجية كثيرا ما تكون عديمة الجدوى وذلك بسبب تنوعها (متطلبات وتحليل وتصميم وخطاطات اختبار) وانفصالها سريعاً عن الواقع بسبب تكرار التغييرات في المتطلبات change requests.

قد يبدو أن تخفيف عدد الوثائق الداخلية إلى وثيقة هي استمارة المتطلبات سيحل المشكلة إلا أن هذه الوثيقة تتضمن حقول صورية يفهمها البرنامج وتتضمن حقول غير صورية يفهمها الإنسان وتسمح بسهولة بالتعبير عن دلالة الحقول الصورية. فمثلاً إذا ربطنا حقل الراتب بمعادلة حساب فقد يكون من الصعب فهم المعادلة من تحليلها إلى جداءات ومجاميع وربما يكون مضيعة للوقت لذلك تتضمن استمارة المتطلبات مقابل كل حقل صوري حقلاً نصياً يشرح دلالة الحقل الصوري. وبما أن بيئة العمل لا تفسر هذا الحقل فيمكن للمهندس أهماله وتجاوزه تحت ضغط العمل مما يجعل الاستمارة قابلة للفهم من قبل المهندسين فقط ويؤثر على سهولة فهمها من قبل المحللين الوظيفيين ومن قبل الزبون.

لذلك قامت الإكسير بما يلي

1- دمج وثيقة تحليل المتطلبات بالمساعدة online بحيث أن إضافة أي مساعدة online من قبل محلل النظم على النظام النهائي سينعكس آلياً على وثيقة المتطلبات ويقوم بتحديثها وبالتالي فلا يوجد حاجة للتنسيق بين محلل النظم الذي يحدث الجانب الدلالي من الوثيقة والمهندس الذي يحدث الجانب الفني منها
2- سمحت الإكسير بتطوير وثائق المتطلبات بشكل تفاعلي أثناء استخدام النظام ودون الحاجة للعودة للوثيقة الأصلية ويمكن في أي لحظة تصدير هذه الوثيقة التي ستتضمن البعد الفني والدلالي بآن واحد
3- إيجاد طباق صريح بين واجهات الإستعمال والتوصيف. في الواقع فإن المهندس الذي يجيد استعمال برمجية ما من الإكسير فهو بالضرورة يجيد برمجتها وتعديلها أيضاً لأنه يستطيع معرفة العلاقة بين الرماز والبرنامج بدقة 100% والسبب هو استعمال مكونات تحليل وتصميم جاهزة أثناء بناء هذه البرمجيات باستعمال بيئة عمل الإكسير. وقد خفف هذا من مخاطر تعاقب مهندسين مختلفين على البرمجيات أثناء فترة حياتها الطويلة نسبياً أسوة ببقية الصناعات الهندسية مثل صناعة السيارت مثلاً.

5.5 الانتقال وممانعة التغيير

توجد العديد من الدراسات عن أسباب ممانعة التغيير **resistance to change** في حالة البرمجيات. فمنها ما يعود لأن الموظف اعتاد برمجيات معينة ولا يريد العودة للتعلم من جديد أو أنه يخشى منافسة الجيل الشاب له في استعمال البرمجيات الحديثة أو أنه يخشى استبداله بهذه البرمجيات وربما لفساد إداري معين.

كما قد يكون **الإنتقال** صعباً بسبب صعوبة تهجير المعطيات القديمة والتدرب على الأنظمة الجديدة والممانعة التي تزداد مع تأخر الانتقال إلى النظام الجديد. في الواقع فالانتقال من برمجيات قديمة إلى برمجيات جديدة هو عملية حساسة للغاية. حتى في حال وجود برمجيات مؤسساتية مختبرة ومستقرة ومجربة مثل أنظمة إدارة المشافي HMIS فإن مجرد

تشغيلها في مشفى جديد يعتبر عملاً شاقاً جداً يتطلب عدة سنين ويمر بصعوبات كثيرة منها تهجير المعلومات القديمة للنظام الجديد والتدريب والتشغيل

ولذلك قامت الإكسير بما يلي:

1- طورت **مسرى مؤسساتي** Enterprise Bus وآليات تهجير جديدة تسمح بالبدء باستعمال النظام حتى قبل الإنتهاء من تهجير المعطيات السابقة وتهجيرها لاحقاً. سمح ذلك للمستخدمين من التمكن من المنظومة قبل تهجير معطياتهم لها والقيام بعملية التهجير بأنفسهم مما يخفض من تكاليف عملية التهجير
2- بالنسبة للتدريب فإن الإكسير **اعتمدت مبدأ الأتمتة 100%** أي أن المستخدم لا يقوم بإدخال إلا المعلومات الخام والموجودة على الأوراق أمامه. فلا يحتاج أي مستخدم عند إدخال فاتورة يومية اعتيادية أو عقد بيع أو شراء أو حتى مؤونة إلى فهم الشجرة المالية أو الأوامر المستودعية.
3- قامت أيضاً بتطوير **online help** لدعم المستخدم أثناء العمل.

6 مراحل استعمال الإكسير

توضح الفقرة التالية المراحل المختلفة لاستعمال بيئة الإكسير في حين تعطي الفقرة التي تليها مثال تطبيقي.

6.1 مراحل استعمال الإكسير

تمر عملية بناء منظومة مؤسساتية باستعمال الإكسير بالمراحل التالية

6.1.1 تحليل المنظومة

وهنا نقوم ببناء النموذج التحليلي CIM أو Computational Independent Model

6.1.2 الانتقال إلى نموذج صوري

وهنا نقوم بتحويل CIM إلى PIM أي Platform Dependant Model وفقا للمراحل التالية

6.1.2.1 تعريف النماذج

حيث يتم تعريف الأغراض الوظيفية مثل موظف أو عملة أو مستخدماً وعملية بيع أو شراء أو غيرها وتعريف خصاص كل غرض وعلاقاته من خلا استمارة جاهزة ويمكن للإكسير إظهار التعريفات بصيغة UML.

6.1.2.2 تعريف الإجراءات

وهنا نقوم بتعريف إجراء وإضافة الخطوات المختلفة من خلال استمارة خاصة بذلك. مثال

6.1.2.3 نماذج أخرى للمكاملة مع خدمات أخرى

الأعمال الخلفية والمهام والمؤقتات والآثار الوظيفية و الأمن وغيرها.

6.1.3 تحديث قاعدة البيانات

في هذه المرحلة نقوم آلياً بإنشاء الجداول الموافق للنماذج ضمن قاعدة البيانات

6.1.4 استعمال المنظومة

بمجرد الانتهاء والضغط على زر التسجيل سوف تتبدل الواجهات تظهر الخمات الجديدة وتكون جاهزة للاستعمال

6.1.5 ما الذي سنحصل عليه بعد النشر؟

1- قاعدة بيانات مع جميع عمليات القراءة والكتابة منها
2- واجهات الاستعمال مع المساعدة online
3- الخصائص غير الوظيفية المذكورة في الفقرة 6

6.2 مثال: نظام إدارة الحجوزات الفندقية

لنأخذ للشرح كمثال متكامل من نظام الإكسير:

أولاً نقوم بكتابة التحليل للنظام الحالي CIM (Computation Independent Model) ثم بتحويله إلى ال PIM (Platform Independent Model)

6.2.1 CIM

الفقرات التالية تعبر عن البنود الأساسية لكتابة CIM بحيث يتم استخدام كلمات مفتاحية تساعدنا على الانتقال آلياً وبسهولة إلى الـPIM عبر مطابقة هذه الكلمات مع مايقابلها في النموذج الصوري PIM

6.2.1.1 *مقدمة*

تحتاج أغلب الفنادق لطريقة فعالة لتنظيم عمليات حجز الغرف ودخول زائر وخروجه لكي يتم استثمار جميع الغرف وتحقيق أفضل نسبة امتلاء. إلا أن هذه العملية تصبح أكثر تعقيداً وعرضة للأخطاء في حال وجود عدد كبير من الغرف والحجوزات وعدد كبير من الموظفين الذين يقومون بحجزها مما قد يتسبب إما ببطء في عملية الحجز للتحقق من الغرف الفارغة أو حجز متكرر مما يسيء للعلاقة مع الزبون.

يهدف هذا النظام إلى تنظيم عملية الحجز بوجود عدد من الموظفين بما يضمن سرعة في عملية الحجز وتجنب للحجز المتكرر وضمان لتأكيد الحجز والفوترة بشكل صحيح.
يتضمن النظام حالات الاستعمال التالية:

- تعريف نوع غرفة داخل الفندق مثل شاليه أو غرفة أو جناح أو غيره.
- تعريف غرفة معينة برقم معين ومبنى خاص ومعلومات أخرى وإلى القيام بعملية طلب حجز لغرفة بوقت محدد ومدة معينة حيث تظهر له الغرف المتاحة في هذه المدة والحجوزات السابقة على هذه الغرف في هذه المدة.
- تأكيد حجز ما بشكل مؤقت بعد التأكد من المدة والمتاحية ولا تسمح بوجود تضارب بين حجزين لنفس الغرفة.
- تأكيد حجز بشكل نهائي بعد التأكد ومن وجود دفعة مقدمة من الشخص المستأجر لهذه الغرفة.
- تسجيل دخول لغرفة ما لزائر معين لتتحول حالة الغرفة إلى مشغولة.
- تسجيل خروج زائر لغرفة لتتحول حالة الغرفة إلى شاغرة.

6.2.1.2 *الخدمات*

وتتضمن قائمة بالخدمات (أسماء الإجراءات) التي سيقدمها النظام مثل الموارد البشرية/التوظيف أو الموارد البشرية/الاستقالة، أو الأرشفة/ إضافة وثيقة جديدة أو الأرشفة/تعديل وثيقة وهكذا.

في المثال: ما هي الخدمات المطلوبة:

1. تعريف نوع غرفة جديد
2. تعريف غرفة جديدة
3. تقديم طلب حجز
4. تثبيت حجز بشكل جزئي
5. تثبيت حجز بشكل كامل
6. إلغاء حجز

7. تعديل حجز
8. عمليةcheck-in
9. عملية check -out

6.2.1.3 *سير العمل*

وفيه وصف مبسط لسير إجراء العمل المتوقع لتقديم خدم معينة و من أجل كل خدمة نضع كافة العمليات اللازمة لهذه الخدمة ونستخدم الكلمات المفتاحية لكل نوع عملية ومعناها:

لنشرح اول خدمتين:

6.2.1.3.1 تعريف نوع غرفة جديد:

يقوم مستخدم مخول باختيار "أنواع الغرف" من المدخل الخدمات الفندقية/إعدادات مما يسمح له باستعراض أنواع الغرف المعرفة سابقاً حيث تظهر الخصائص التالية:

1. معرف نوع الغرفة
2. اسم نوع الغرفة

والبحث ضمنها وفقاً لما سبق من خصائص مع إمكانية إظهار تفصيلي لنوع الغرفة وتعديلها.
مع إمكانية تعريف نوع غرفة جديدة وإدخال خصائصها.

6.2.1.3.2 تعريف غرفة جديدة:

يقوم مستخدم مخول باختيار "إدارة لغرف" من المدخل الخدمات الفندقية مما يسمح له باستعراض الغرف المعرفة سابقاً حيث تظهر الخصائص التالية:

1-رقم الغرفة

2-نوع الغرفة

3-سعر الغرفة

4-المبنى الخاص بالغرفة

5-الزائر

6-حالة الغرفة (شاغرة أم لا) – *هذه الواصفة تساعدنا من أجل عملية تسجيل دخول زائر أو تسجيل خروجه* -

والبحث ضمنها وفقاً لجميع ما سبق من 1 إلى 4 بالإضافة إلى البحث بمن وإلى تاريخ للبحث عن الغرف المتوفرة أو غير المتوفرة بين تاريخين .

كما يستطيع المستخدم من خلال الضغط بالزر اليميني على غرفة ما اختيار ما يلي من القائمة المنسدلة التي تظهر:

- "إظهار" يتم الانتقال الى واجهة تظهر فيها جميع المعلومات أعلاه من 1 إلى 4
- "تعديل" يتم الانتقال إلى واجهة لتعديل معلومات الغرفة
- "عملية check-in" يتم الانتقال إلى واجهة لإدخال زائر معين إلى الغرفة
- "عملية check-out"يتم الانتقال إلى واجهة لتسجيل خروج زائر من الغرفة.
- "الحجوزات السابقة"يتم الانتقال إلى واجهة أخرى تسمح له باستعراض جميع الحجوزات السابقة لهذه الغرفة بين تاريخين مع إمكانية إنشاء حجز جديد لهذه الغرفة وفيها يتم الانتقال إلى واجهة لإدخال خصائص الغرفة

ويظهر للمستخدم ثلاثة أزرار لحفظ هذا الحجز إما "حفظ" فيقوم بحفظ الحجز بحالة طلب حجز دون تثبيته

أو " تثبيت جزئي" فيتم حفظ الحجز بحالة حجز مؤقت إلى أن يتم إضافة دفعة مقدمة عليه

أو "تثبيت دائم" فيتم حفظه بحالة حجز دائم ويجب إضافة دفعة مقدمة له.

6.2.1.3.3 تقديم طلب حجز/ تثبيت حجز بشكل جزئي أو كامل / تعديل أو إلغاء حجز:

6.2.1.3.4 عملية Check-In/out لغرفة معينة.

6.2.1.4 *الأغراض:*

ونذكر هنا قائمة بالأغراض الوظيفية التي تتعامل معها الإجراءات السابقة.

اسم الغرض <الأول> نوع غرفة

حقوق الاستعراض viewing: < نضع هنا وصف لمن يستطيع رؤية الغرض وبأية شروط ونستنتج ذلك من إجراءات العمل السابقة أي أن حقوق الاستعراض هي منظور view آخر لإجراءات العمل. كما في المثال التالي >

- يرى الجميع أنواع الغرف المعرفة مسبقاً من النظام ويمكن تعريف نوع غرفة جديد لشخص مخول يحدده النظام لاحقاً

<اسم الغرض الثاني>: غرفة
حقوق الاستعراض:

- يرى موظف الحجز الغرف المعرفة مسبقاً ويستطيع تعريف غرفة جديدة ورؤية الغرف المتاحة لإنشاء حجز معين بين تاريخين.
- يرى موظف الاستقبال المسؤول عن تسجيل دخول زائر معين إلى غرفة ويسجل دخول أو خروج زائر إلى غرفة معينة.

<اسم الغرض الثالث>: حجز
حقوق الاستعراض:

- يرى موظف الحجز الحجوزات السابقة كلها ويقوم بتثبيت حجز بشكل مؤقت أو دائم أو إلغاء أو تعديل.

6.2.2 PIM

وهو توصيف صوري للنظام لا يقبل التأويل ويتكون من مجموعة من الاستمارات الخاصة بالنماذج المراد توصيفها ومنها

- نموذج الغرض: وصف لكل حقل من الغرض.
- نموذج إجراء العمل: ويتضمن مجموعة من الخطوات بأنواع مختلفة مثل المهام الآلية واليدوية والأعمال الخلفية وغيرها
- نماذج تصريحية لمكاملة هذه الخدمة مع الخدمات الأخرى الوظيفية والتقنية.

6.2.2.1 *نوع الغرفة*

سنأخذ مثال أول غرضين:

6.2.2.1.1 نموذج الغرض

الرمز	الوصف
id	المعرف المميز الخاص بكل نوع غرفة
name	اسم نوع الغرفة

6.2.2.1.2 العمليات على الغرض

الرمز	الشروط المسبقة	الوصف
setting	-	إعدادات الغرف
New	-	نوع غرفة جديد
View	-	إظهار
Edit	-	تعديل

6.2.2.2 *الغرفة*

6.2.2.2.1 نموذج الغرض:

الرمز	الوصف
id	المعرف المميز الخاص بكل غرفة
roomType	نوع الغرفة وتكون قيمه من أنواع الغرف المعرفة مسبقاً
entity	المستأجر الذي يشغل الغرفة
build	وهو المنشأة (الفندق) الذي تكون فيه الغرفة
price	وهو سعر الغرفة في اليوم

6.2.2.2.2 العمليات على الغرض

الرمز	الشروط المسبقة	الوصف
Room Mangment	-	إدارة الغرف
New	-	غرفة جديد
View	-	إظهار
Edit	-	تعديل
reservations		الحجوزات السابقة
check-in	entity == null	تسجيل دخول
checkout	entity!=null	تسجيل خروج

6.2.2.3 *الحجز ..*

6.2.2.4 وصف نموذج الأدوار كنموذج تكامل مع خدمة الأمن:

نضع هنا وصف لكل دور وما هي صلاحياته ونستنتج ذلك من إجراءات العمل السابقة أي أن وصف الأدوار هو منظور view آخر لإجراءات العمل

الدور بالعربي	الدور بالإنكليزي	الوصف
موظف الاستقبال	receptionist	وهو المسؤول عن إدخال معلومات الزبائن وتسجيل دخولهم وخروجهم لكل غرفة.
موظف الحجز	Reservation employee	وهو المسؤول عن القيام بالحجوزات.
المحاسب	accountant	وهو المسؤول عن القيام بالعمليات المالية واستلام دفعة مقدمة من الزبائن.

6.2.2.5 وصف الصلاحيات:

الأدوار الواردة أعلاه لا تعكس الهرمية الإدارية ولا المناصب الوظيفية في المؤسسة لذلك نقوم بتجميع صلاحيات هذه الأدوار ضمن قوالب لتخفيف عدد الصلاحيات ثم نربط القوالب بالمناصب الوظيفية المناسبة.

يتم تحديد القوالب من خلال مراجعة السيناريوهات الوظيفية

يجب تجنب ربط الصلاحيات مع المستخدمين مباشرة لان ذلك سيؤدي لدرجة تعقيد كبيرة ففي حال كان لدينا 1000 صلاحية و100 مستخدم سيؤدي ذلك إلى عمل من رتبة 100 ألف إسناد لذلك نقوم بتجميع الصلاحيات في حوالي 20 قالب ونجمع المستخدمين بحوالي 10 مناصب ثم نسند القوالب للمناصب مما يعطي درجة تعقيد من رتبة 200 بدلا من 100 ألف.

6.2.3 الانتقال إلى الإكسير:

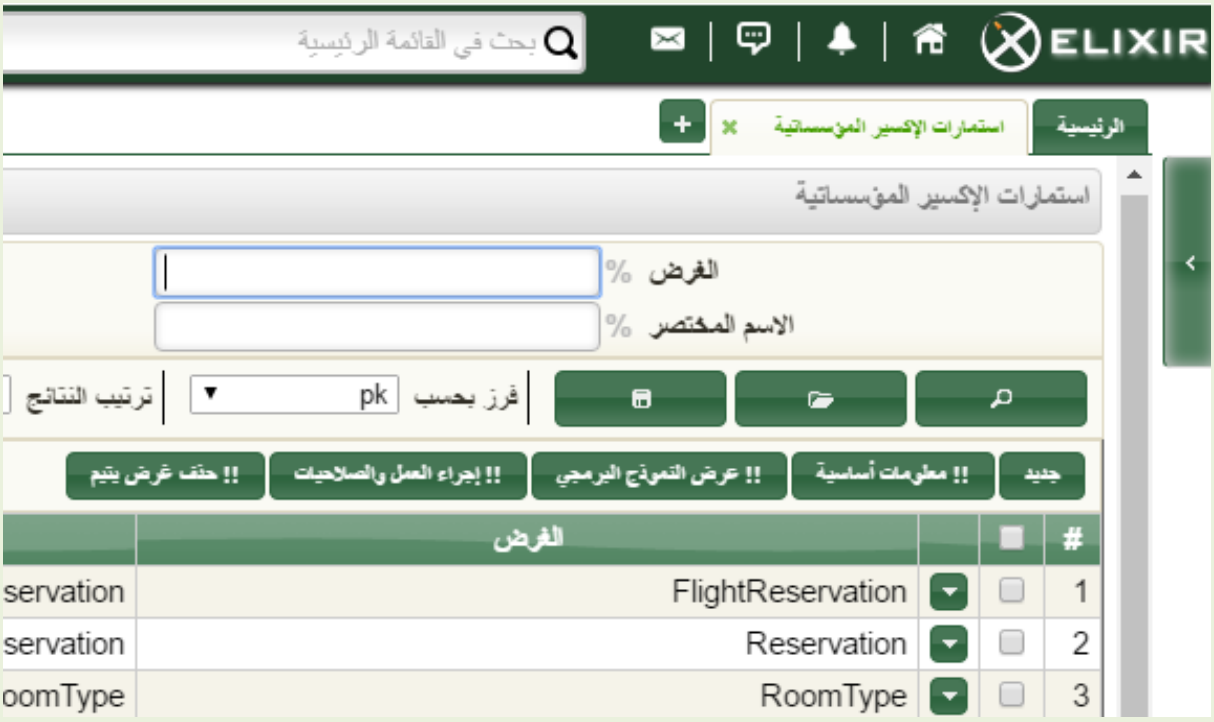
1- إرساء بيئة عمل الإكسير على الحاسب.

2- يمكن استيراد الـPIM إلى الإكسير وتشغيل النظام فوراً إذا كان مكتوب بالبنية المعيارية للإلكسير بصيغة xml ولكن لصعوبة كتابتها يدوياً يوجد مداخل من الإلكسير تسمح بإدخالها خطوة خطوة ويمكن أيضاً تصديرها لسهولة استخدامها من نسخة لأخرى.

لبناء النموذج بمساعدة واجهات تفاعلية في بيئة عمل الإلكسير:

- في حال كان المستخدم الذي دخلت منه إلى الإكسير لديه الصلاحيات لإدارة الوثائق الإلكترونية يمكن الدخول وتعريف الأغراض والحقول السابقة مع علاقاتها من المدخل إدارة الوثائق الإلكترونية ← النمذجة ← استمارات الإلكسير المؤسساتية

ولتعريف صف جديد ← جديد



لتعريف صف معين في الإكسير لدينا نموذج للغرض Class Template

في المثال سنعرف الغرض "نوع الغرفة " كما يلي:

ولتعريف حقول الغرض ندخل إلى إدارة الحقول:

ومن ثم إضافة حقل جديد:"حقل اسم نوع الغرفة "

لتعريف حقل معين في الإكسير لدينا نموذج للحقل Field Template:

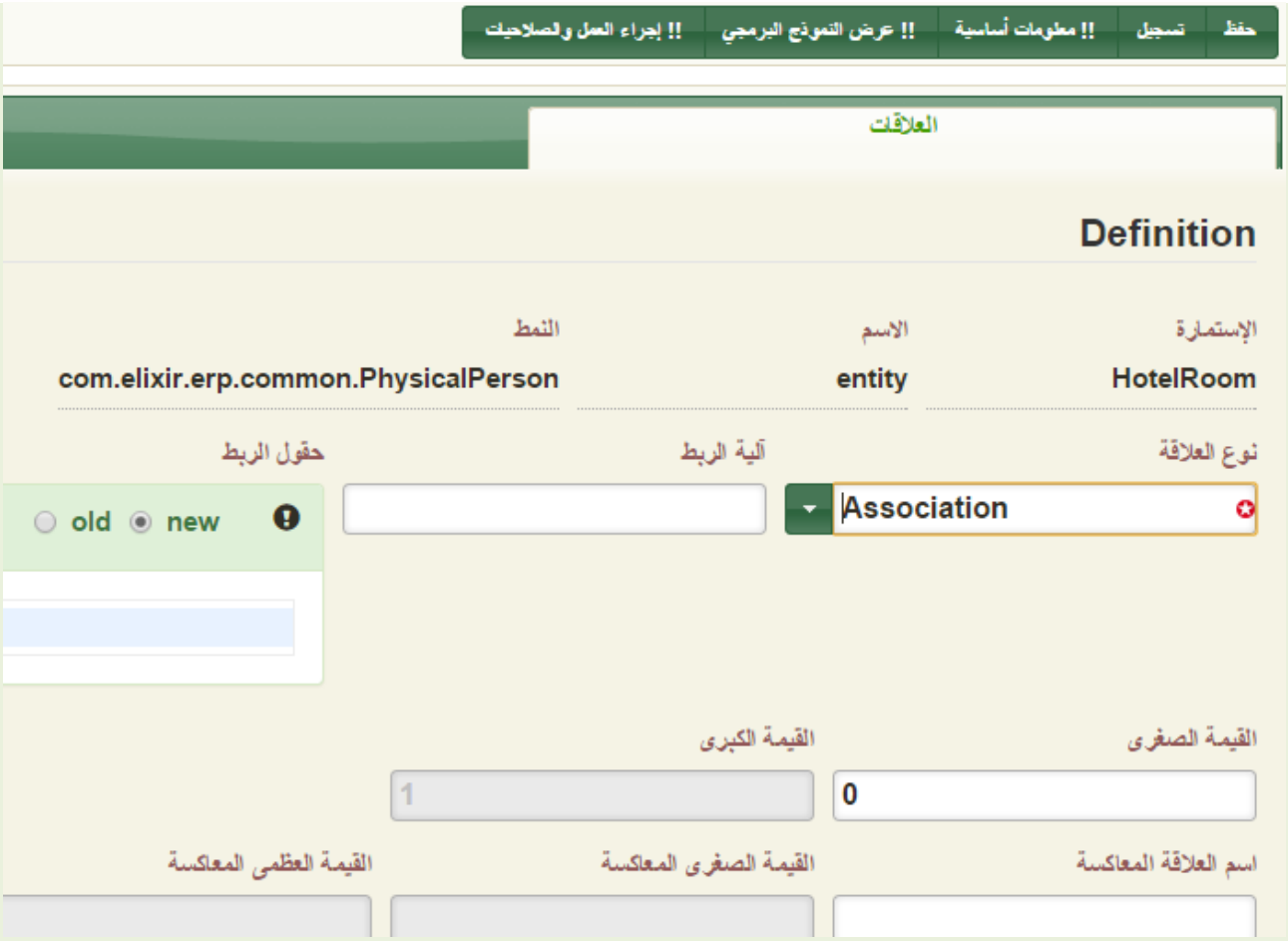
وهكذا من أجل بقية الحقول.

ثم لتعريف الغرض الثاني وهو الغرفة نقوم بنفس العمل لتعريف الصف ثم الحقول.

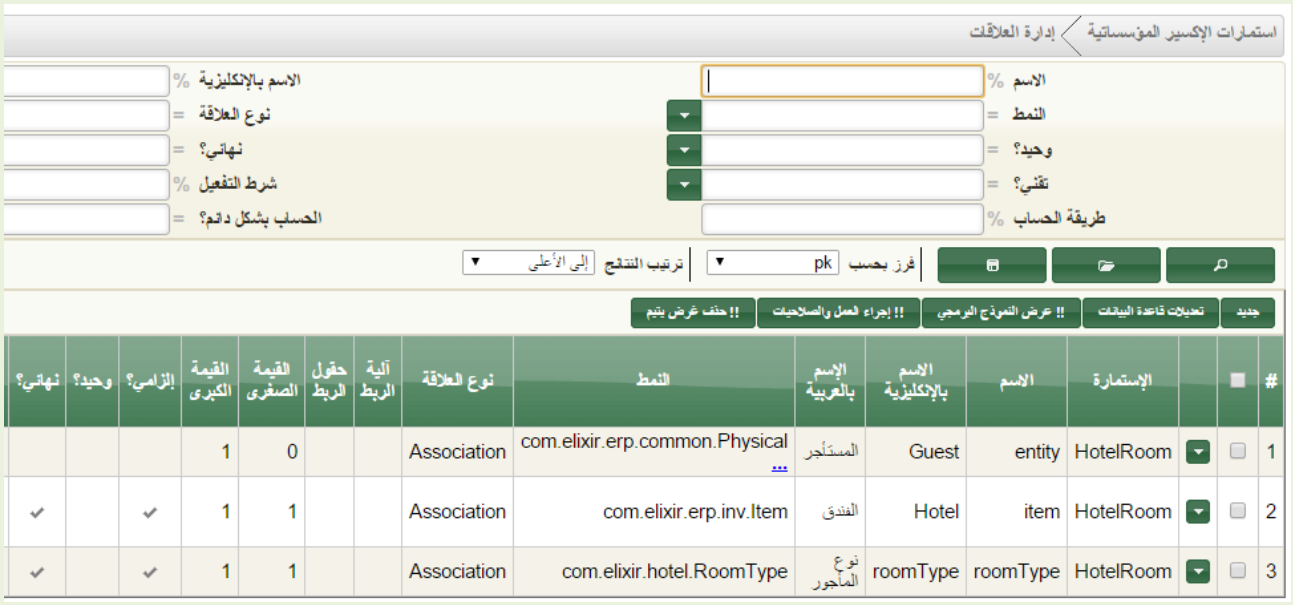
ولكن لدينا الواصفة:" نوع الغرفة، المستأجر، المنشأة" هي علاقة من نوع association مع الغرفة لذا نفتح "إدارة العلاقات" لتعريف علاقات للصف الحالي

ولتعريف علاقة معينة في الإكسير لدينا نموذج للعلاقة Relation Template:

نعرف علاقة مع الغرض "نوع الغرفة ":


حفظ
تسجيل
!! معلومات أساسية
!! عرض النموذج البرمجي
!! إجراء العمل والصلاحيات
العلاقات
Definition
الإستمارة
HotelRoom
الاسم
entity
النمط
com.elixir.erp.common.PhysicalPerson
نوع العلاقة
Association
آلية الربط
حقول الربط
old
new
القيمة الصغرى
0
القيمة الكبرى
1
اسم العلاقة المعاكسة
القيمة الصغرى المعاكسة
القيمة العظمى المعاكسة

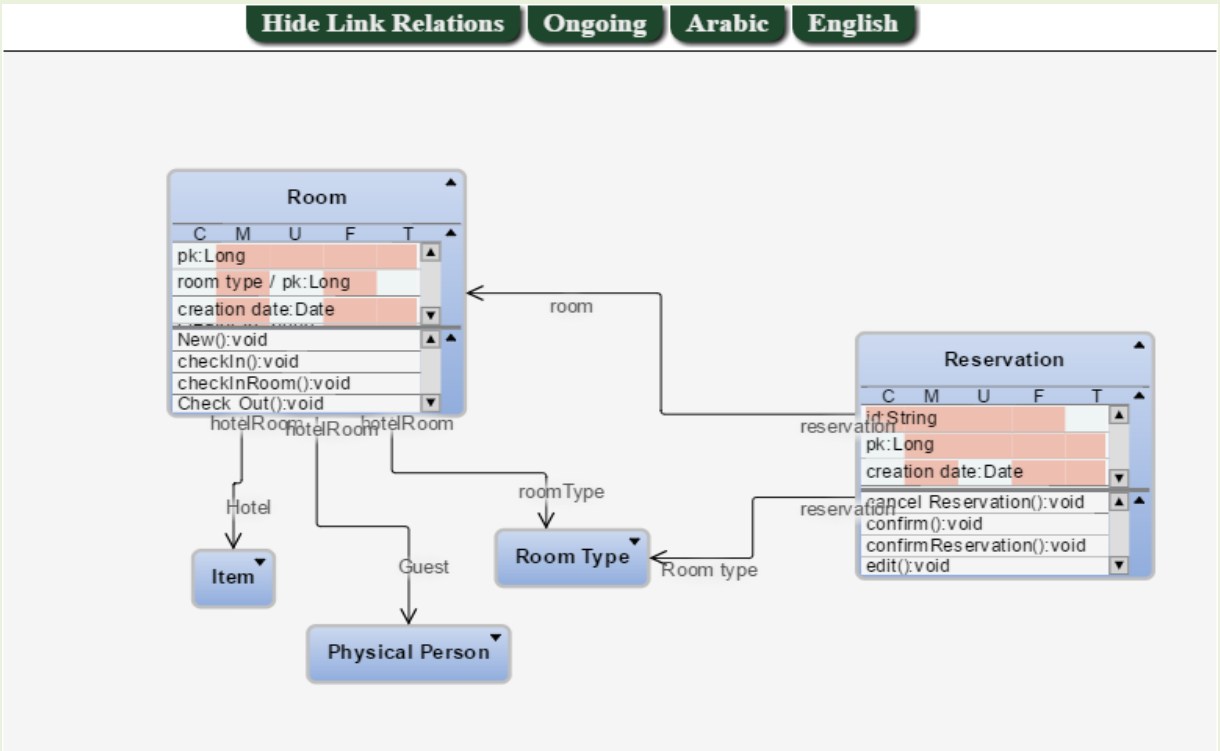
نلاحظ أيضاً إمكانية الربط مع أغراض سابقة في نظام الإكسير مثل المنشأة من نوع item والمستأجر من نوع Entity لذا تكون العلاقات في هذا الصف:



#		الإستمارة	الاسم	الاسم بالإنكليزية	الإسم بالعربية	النمط	نوع العلاقة	آلية الربط	حقول الربط	القيمة الصغرى	القيمة الكبرى	إلزامي؟	وحيد؟	نهائي؟
1		HotelRoom	entity	Guest	المستأجر	com.elixir.erp.common.Physical ...	Association			0	1			
2		HotelRoom	item	Hotel	الفندق	com.elixir.erp.inv.Item	Association			1	1	✓		✓
3		HotelRoom	roomType	roomType	نوع المأجور	com.elixir.hotel.RoomType	Association			1	1	✓		✓

4-إنشاء جدول لهذه الأغراض في قاعدة البيانات من خلال الضغط على زر "إنشاء جدول قاعدة البيانات".

5-تسجيل هذا الصف في قاعدة البينات من خلال الضغط على زر "تسجيل".



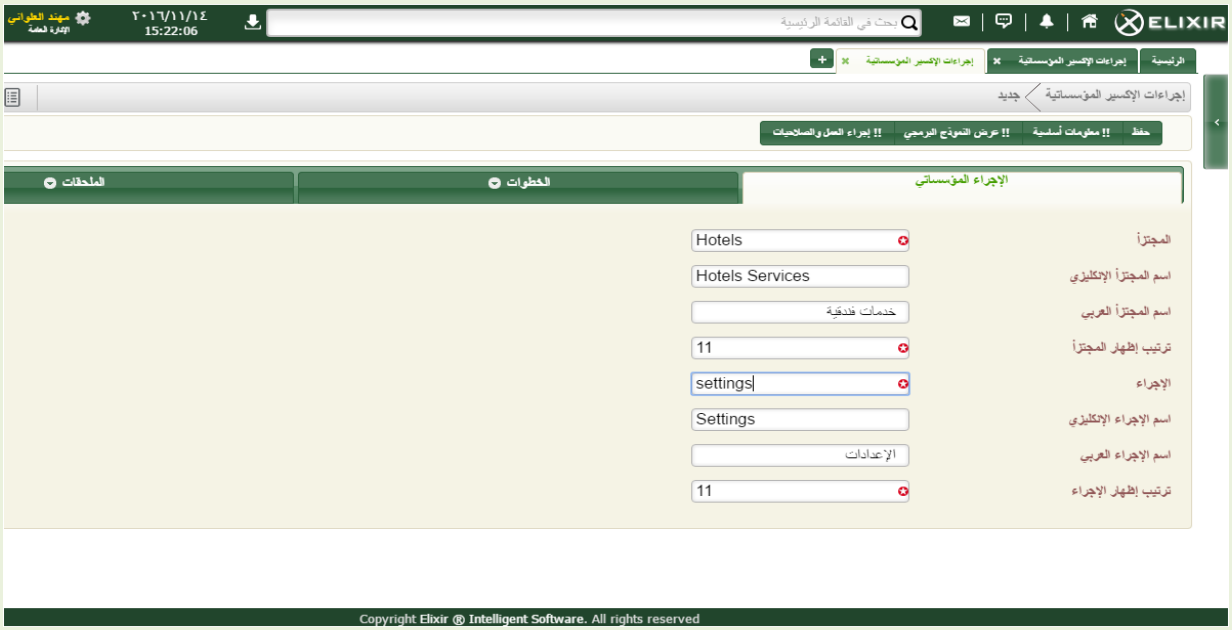
ويكون لدينا النموذج الحالي UML بعد إكمال بقية الصفوف كما يلي:

6-بناء الخطوات المطبقة على هذا الغرض من خلال المدخل إدارة الوثائق الإلكترونية ← النمذجة ← إجراءات الإلكسير المؤسساتية

لتعريف إجراء معين في الإكسير لدينا نموذج للإجراء Workflow Template:

لدينا في النظام الحالي إجرائيين رئيسين وهما الإعدادات (وفي يتم تعريف أنواع غرف) والحجوزات الفندقية وفي كل منهما عدة خطوات

الإجراء "إعدادات ":



لتعريف خطوة معينة لإجراء في الإكسير لدينا نموذج للخطوة Step Template:

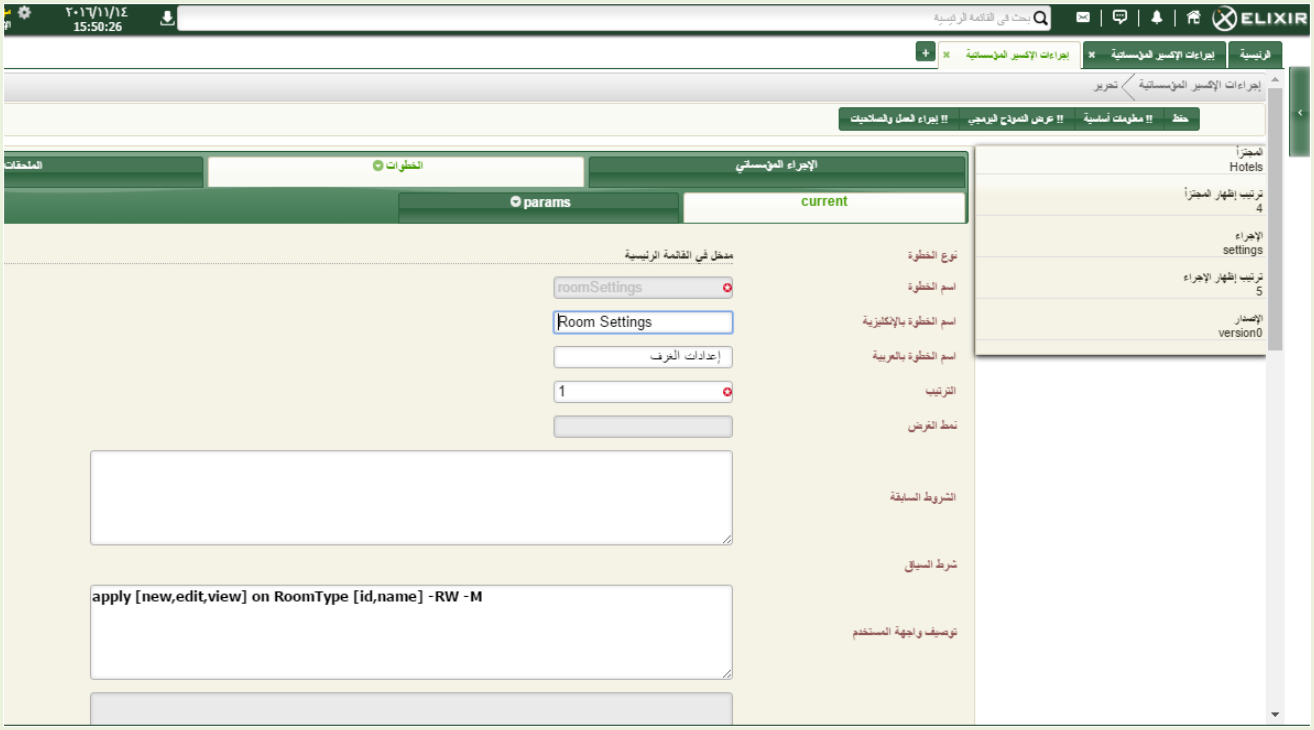
تعرف الخطوات عند الضغط على الـ tab المسماة "الخطوات"

يوجد أنواع للخطوات في الإكسير منها ما هو مداخل في القائمة الرئيسية ومهمات
ومنها أعمال خلفية للنظام Job يقوم بها كل فترة محددة.
أو مهام آلية يقوم بها النظام لوحده أو يدوية يقوم بها دور معين عند تحقق شروط معينة.
أو أعمال تابعة لأغراض معينة أو أعمال مستقلة.

- بما أن واجهات الإكسير تولد آلياً فبعض هذه الخطوات هو عبارة عن انتقال من واجهة إلى أخرى كخطوة (جديد) التي تقوم بالانتقال من واجهة تعرض كل الغرف المعرفة مسبقاً إلى واجهة تسمح لي بتعريف غرفة جديدة وبعض هذه الخطوات هي عبارة عن أعمال تطبق دون انتقال بين الواجهات كـ(حفظ) أو (تثبيت مؤقت لحجز).

لدينا الخطوات في الإجراء "الإعدادات":

1-إعدادات الغرف وهي من نوع مدخل في القائمة الرئيسة لأنها مدخل يعرض من خلاله كل أنواع الغرف ويمكن من خلاله



تعريف نوع غرفة جديد.

ويعرض فيها العمليات " جديد" كزر يمكن منه تعريف نوع غرفة جديد و "عرض أو تعديل "عند الضغط بالزر اليميني على أي نوع غرفة لذا تم تطبيق الخطوات new،view، edit على الصف RoomType ويظهر من أنواع الغرف المعروضة الحقلين "id، name"

ولكن عند تسجيل الخطوات نلاحظ أن العملية جديد موجودة كعملية مستقلة في الواجهة أما عرض وتعديل تظهر عند الضغط على الزر اليميني في القائمة المنسدلة لكل نوع غرفة:

2- جديد لتعريف نوع غرفة جديد وهي من نوع عمل يدوي مستقل لأنها خطوة static على الصف RoomType تنتقل إلى واجهة تسمح لي بتعريف نوع غرفة جديد.

3- تعديل لتعديل نوع غرفة وهي من نوع عمل يدوي تابع لغرض لأنها تطبق على غرض معين من نوع "نوع غرفة" وتنقلني إلى واجهة تسمح لي بتعديل معلومات الغرفة:

4- عرض لعرض معلومات نوع الغرفة وهي من نوع عمل يدوي تابع لغرض وهذه الخطوة أيضاً تشبه الخطوة السابقة لكن دون تطبيق أي عملية وتكون RO- أي read only:

بعد الانتهاء من هذه الإجراء وكافة خطواته نحفظ ونقوم بتسجيله.

7- تسجيل الإجراء.

8- لرؤية الإجراء وخطواته في النظام يجب إعطاء المستخدم الحالي الصلاحيات لإظهاره فيصبح الإعداد موجود ضمن القائمة الرئيسية ويمكن الوصول لكل خطواته من داخل النظام.

ومن أجل الإجراء الثاني نقوم بنفس الطريقة ولدينا الخطوات التالية:

1- إدارة الغرف:

وهي من نوع مدخل في القائمة الرئيسة لأنها مدخل يعرض من خلاله كل الغرف ويمكن من خلاله تعريف غرفة جديدة.

2- جديد غرفة

لتعريف غرفة جديدة وهي من نوع عمل يدوي مستقل لأنها خطوة static على الصف RoomHotel أي أن هذه الخطوة غير متعلقة بأي غرض من هذا الصف.

3- عرض غرفة

لعرض معلومات الغرفة وهي من نوع عمل يدوي تابع لغرض

4- تعديل غرفة

لتعديل معلومات غرفة وهي من نوع عمل يدوي تابع لغرض لأنها تطبق على غرض معين من نوع "غرفة"

5- الحجوزات السابقة لغرفة.

لعرض الحجوزات السابقة لغرفة وهي من نوع عمل يدوي تابع لغرض وتسمح لي بالانتقال إلى واجهة تحوي كافة الحجوزات السابقة لهذه الغرفة بين تاريخين ويمكن منها تعريف حجز جديد لهذه الغرفة.

6- تسجيل دخول لغرفة.
لإدخال زائر معين لهذه الغرفة وهي أيضاً من نوع عمل يدوي تابع لغرض لأنها متعلقة بالغرفة المراد إدخال زائر لها وتقوم بالانتقال إلى واجهة تسمح بذلك

7- تسجيل خروج لغرفة

لإخراج زائر معين من هذه الغرفة وهي من نوع عمل آلي لأنها تقوم بإخراج الزائر الموجود بالغرفة بعد التأكد من أن ليس للزائر حساب لم يتم دفعه.

وهكذا من أجل بقية الخطوات الخاصة بالحجز.

Elixir for Intelligent Software (L.L.C)
Damascus – Syria
Phone : +963 11 2149 1991
Fax: +963 11 2149 1990
web: www.el-ixir.com
E-mail: info@el-ixir.com